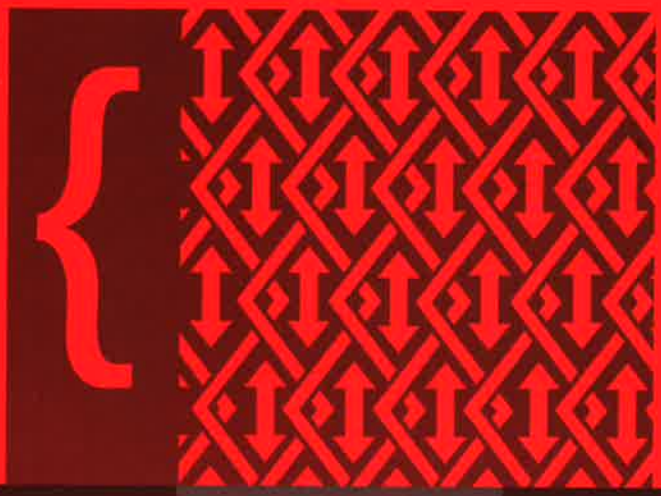
;login:

THE MAGAZINE OF USENIX & SAGE

December 2001 • Volume 26 • Number 8



inside:

This is how one pictures the angel of history. . . . Where we perceive a chain of events, he sees one single catastrophe which keeps piling wreckage upon wreckage and hurls it in front of his feet. The angel would like to stay . . . and make whole what has been smashed. But a storm is blowing from Paradise . . . . This storm irresistibly propels him into the future to which his back is turned. . . . This storm is what we call progress.

Walter Benjamin, Theses on the Philosophy of History, IX.

USENIX & SAGE

The Advanced Computing Systems Association & The System Administrators Guild

USENIX Upcoming Events

Conference on File and Storage Technologies (FAST)

Sponsored by USENIX, Co-sponsored by IEEE TCOS, in cooperation with ACM SIGOPS

JANUARY 28–30, 2002 Monterey, California, USA

http://www.usenix.org/events/fast/

BSDCon 2002

FEBRUARY 11–14, 2002 San Francisco, California, USA

http://www.usenix.org/events/bsdcon02/

The 4th NordU/USENIX Conference (NordU/USENIX 2002)

Co-sponsored by USENIX and EurOpen.SE–The Swedish Association of UNIX Users

FEBRUARY 18–22, 2002 Helsinki, Finland

http://www.nordu.org/NordU2002

The 3rd International SANE Conference

Organized by NLUUG
Co-sponsored by USENIX and the NLnet Foundation.

MAY 27–31, 2002 Maastricht, The Netherlands

http://www.sane.nl

2002 USENIX Annual Technical Conference

JUNE 9–14, 2002
Monterey, California, USA

http://www.usenix.org/events/usenix02/

2nd Java™ Virtual Machine Research and Technology Symposium (JVM '02)

AUGUST 1–2, 2002 San Francisco, California, USA

http://www.usenix.org/events/jvm02

Paper submissions due: February 4, 2002
Notification of acceptance: March 12, 2002
Camera-ready final papers due: May 28, 2002
Registration materials available: April, 2002

11th USENIX Security Symposium

AUGUST 5–9, 2002 San Francisco, California, USA

http://www.usenix.org/events/sec02

Paper submissions due: January 28, 2002

16th Systems Administration Conference (LISA '02)

Sponsored by USENIX & SAGE

NOVEMBER 3–8, 2002 Philadelphia, Pennsylvania, USA

2nd Workshop on Industrial Experiences with Systems Software (WIESS '02)

Sponsored by USENIX
Co-sponsored by ACM SIGOPS, & IEEE TCOS

DECEMBER 8, 2002 Boston, Massachusetts, USA

http://www.usenix.org/events/wiess02

Submissions due: July 15, 2002

5th Symposium on Operating Systems Design and Implementation (OSDI '02)

Sponsored by USENIX
Co-sponsored by ACM SIGOPS, & IEEE TCOS

DECEMBER 9–11, 2002 BOSTON, MASSACHUSETTS, USA

http://www.usenix.org/events/osdi02

Submissions due: May 24, 2002

contents

;login: Vol. 26 #8, December 2001

;login: is the official magazine of the USENIX Association and SAGE.

;login: (ISSN 1044-6397) is published bimonthly, plus July and November, by the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

$50 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $60 per year.

Periodicals postage paid at Berkeley, CA, and additional offices.

POSTMASTER: Send address changes to *;login:*, USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

motd

by Rob Kolstad

Dr. Rob Kolstad has long served as editor of *;login:*. He is also head coach of the USENIX-sponsored USA Computing Olympiad.

kolstad@usenix.org

What's Really Hard

Like all of us, I've been pondering a lot of thoughts lately in the sort of "global" scheme of things. Books like *Guns, Germs, and Steel* and *The Wealth and Poverty of Nations* advance various assertions as to why certain parts of our world have economic and other types of success while others seem always to fall behind. But what's the really hard problem?

I think I've discovered a little bit about what's really hard. I know many things are extraordinarily challenging: being the 2001 home run champion (or all-time home run champion!), basketball MVP, president of the world's biggest company, author of the world's best networking book (or any book, for that matter), programmer of the world's best window system, earner of $100,000 in one year, parent of a child until the child can live on his or her own, and a host of other accomplishments, both private and public.

I think, though, that the really hard challenges are not in doing something once (which is "hard") but in sustaining the performance: five-time basketball MVP, two-term country president, creator of both a revolutionary programming language and an operating system, winning back-to-back Super Bowl championships, earning enough to survive and creating financial security for retirement, paying off the house's 30-year loan, sustaining your operating system through its second batch of thousands of customers, writing a second best-seller, etc.

What does this have to do with all of us? Because we, as a country, have been successful at so very many things: personal liberties (I know there are quibbles, but we do OK), economic success, freedom from fear, plenty of food, plenty of housing, and, best of all, plenty of opportunity. We are by no means the only part of the world to have these things, but we are one part that does. No small part of this success derives from sustaining these values, and that's hard.

That's where all of us come in. As a nation, we must perform the most difficult act in these turbulent times: continuing to sustain the greatness that we, as a country, have built over the last few decades. That means we have to continue to work, grow, build, nest, play, and nurture all the things that have built our society (including dissent and agreement, conflict and harmony, and so on).

Of course, we aren't the only great place – there's plenty of greatness and success to go around the globe – but we need to continue to focus on the work that is required to sustain what we have and grow. Without that focus, I fear we will fall back and lose some of the things that have gone so well in the past.

Sustaining is hard, very hard. Upon reflection, it's the way things have to be. I hope we can all carry on together; it's important.

PS to our foreign readers: I know this is USA-centric; we have a a bit of a national crisis right now. Thanks for your patience, and I hope you'll forgive me for this column.

apropos

Re-Routed Packets

by Tina Darmohray

Tina Darmohray, co-editor of *;login:*, is a computer security and networking consultant. She was a founding member of SAGE.

tmd@usenix.org

On September 11, 2001, I was teaching a tutorial in the Georgia World Congress Center in Atlanta as part of Networld+ Interop. Shortly after the tutorial began, it was interrupted by the news of the hijacks and crashes of several US passenger airplanes. I don't think any of us initially realized the impact of what had happened. For some of us, the events of that day threw a wrench in our travel plans. For others, so much more was changed.

I was originally scheduled to fly out Wednesday on a 6:50 p.m. nonstop to San Jose, California. By Wednesday noon that flight had been cancelled, and I began to try to figure out what else I could do to get home. Several folks in my course offered to let me stay at their houses. One even suggested that I would solve their babysitting problem for back to school night if I came that evening!

Despite the kind offers, I decided it was best to make tangible progress toward California. I knew that Mark Mellis, who lives in Los Angeles, was stranded in Boston and was also trying to sort out his options to get home. Another instructor teaching on Tuesday, Karl Andersen, lives in New Jersey and, like me, had decided in favor of making progress toward home rather than waiting for air travel to start up again. The three of us discussed our options and the probability for success of each (leave it to network security geeks) and cooked up a plan to achieve our common goal of getting home in the midst of the massive travel interruption.

Basically, Karl and I would drive north and meet Mark, who would drive south, to the corner of I-81 and I-70 in Hagerstown, Pennsylvania. From there, Karl would continue home to New Jersey, and Mark and I would head west on I-70, a route that passes through several towns served by Southwest Airlines. Not knowing when air travel would resume or what it would be like when it did, Mark and I thought we might have a better chance of getting on a small carrier out of the Midwest than a larger carrier out of the hub airports. This way, we figured we'd either drive to the Pacific Ocean or catch a plane, whichever came first. What we didn't know at the time is that we'd be sharing this adventure with so many other travelers; middle America was filled with people trying to get home (*http://www.mellis.com/911roadtrip/*).

Judging from my sampling in the continental breakfast areas in the hotels where we stayed, a full 75% of the folks were in the same predicament we were. Some of the people were like us and had been stranded on business or vacation, while others, in the air on Tuesday, had been put down in random locations around the US. In Pennsylvania I talked to a fellow who was in the air on his way to vacation in Las Vegas when his plane had been put down in Indianapolis. He said their captain came on the speaker and said that their plane was OK, but there had been a "national incident" and they had been asked to land. Gosh! I can't imagine what would have been running through my mind with that kind of lead-in but no further details. Since hearing that, I've wondered what folks who listen to the air-traffic channel on the planes heard in real time while this drama was playing out.

The longest drive I heard about was from a couple I talked to in Missouri who had been flying back from a trip to Alaska when their plane landed in Seattle. As it was, they were driving the diagonal from Washington to Orlando, Florida, but they both agreed, it could have been a *lot* farther!

At midday on Friday, Mark and I passed the Indianapolis airport and saw commercial airlines taking off from the runways and jet trails in the otherwise clear sky. It was a welcome sight. A quick cell phone call to Southwest Airlines revealed that we couldn't make their scheduled flight out of St. Louis that evening, but we could get on one out of Kansas City the next morning; we booked it.

We arrived three-and-a-half hours early for our flight out of Kansas City on Saturday. The airport was deserted and security was heightened. Everyone's nerves were frazzled. The ticketing agent who was helping me came out of his skin when he realized that he'd failed to tag the previous customer's luggage, and for an instant, didn't know what to do or think.

Our flight from Kansas City, through Las Vegas to San Jose, was uneventful, save the genuine appreciation by the flight attendants and crew members for us choosing air travel. The flight attendants high-fived me on the way out in San Jose. I've heard similar stories from other travelers. Accounts have it that first-flights into airports were greeted by cheering airport workers out on the tarmac.

My non-scientific sampling of others who tried to get to their destinations the week of September 11 shows that it was harder to get on planes at the large airports. Not being able to get a flight until

apropos (cont'd)

Saturday, Sunday, or even later was relatively common. Amazingly, when folks finally did get off the ground, those flights, which had been so hard to get, turned out to be only two-thirds full. Alternative forms of travel, such as Greyhound, Amtrak, and car rentals, were frequently sold out. I did hear several accounts of companies chartering buses to get groups of co-located employees home with good success (I thought that was pretty clever). I heard from one of my tutorial attendees who lives near me in California that he left from Atlanta Friday night and drove 2,800 miles in three days via Oklahoma and Arizona.

Now that we've gotten back and have had the chance to hear how others did the same, Mark, Karl, and I have applied some hindsight to our quickly concocted plan. Mark believes our initial reaction, and subsequent plan, reflects our collective system/network administration background. He maintains successful system administrators are pragmatic. In the face of building moves, application conversions, and OS upgrades, we know our best chance for success lies in having multiple ways to achieve the goal: built-in disaster planning, if you will. Our experience tells us that occasionally the upgrade goes off without a hitch, but if we positively have to have the site up at the end of the availability window, we know we must also plan for the worst case. That way, when things fall apart midway through the migration, our chances for success are much better, since we planned for other options early on. Of course we'll accept it if things work out for the best, but if we *must* be home on Monday, we rent a car first then check the airports along the way.

letters to the editor

LETTERS FROM USACO FINALISTS

[*Better late than never. The following are some of the letters USENIX received back in July from students who attended the USACO training camp. See also page 94 of this issue and page 92 in the August issue of* ;login:. Ed.]

My name is Songzi Du, and I am one of the 15 USACO finalist this year. I am very grateful for Usenix's sponsoring of USACO camp. The camp brought 15 bright teens from all over the nation together at Wisconsin to think about computer science for a week without any distraction. We learned a lot there. Besides the programming part of the week, I also enjoyed the disc golf and a trip to Chicago. Thank you very much for your sponsoring of USACO.

Songzi

I am writing you to thank you for sponsoring the USA Computing Olympiad, a program in which I have participated for two years. Let me assure of the program's value; in fact, I know that my peers and I consider it to be the premier high school computer science competition as well as the best training program in algorithmic computer science available. Camp this year was a blast, like last year, and the coaches worked very hard, as always, to teach us a lot about the subject material. I'm looking forward to representing the U.S. at the international competition later this month; I think that we will have one of our most competitive teams ever this year, largely thanks to USENIX's support.

Thanks again,

Tom Widland

Thank you very much for sponsoring USACO and making the training camp, which I recently attended, possible!

Anatoly Preygel

I was a finalist at the USA Computing Olympiad. I wanted to thank you for your sponsorship of that organization. It was an excellent experience for me, and I learned a lot. I appreciate your making this possible. Thank you.

Jeff Cohen

Thank you for your continuing support of the USA Computing Olympiad. This year I attended the USACO Training Camp in Wisconsin for the first time, and it was a great experience. The camp's lectures, labs, and competitions offer students an invaluable opportunity to learn about computer science. Without your financial support, this wonderful experience would not have been possible. Thank you.

Jeff Arnold

ON DRM

from Gregory P. Smith
greg@electricrain.com

I must take issue with Daniel Geer's caution against supporting the opposition to digital rights management (DRM) in his column on Electronic Property [October 2001]. The opponents of DRM solutions available from today's slow-moving institutions are doing property based democracies a favor. Insufficient DRM solutions are being exposed for the con-jobs that they are. That should lead to better ones being developed. Unfortunately today's slow-moving institutions seem to spend more on lawyers and politicians. They would rather make it illegal to show how bad their solution is than to spend the same money developing a good solution. A DRM solution in all democratic citizens' best interest will be an open standard involving zero royalties or licensing fees.

more letters . . .

OCTOBER 2001 MUSINGS

From: Murray Stokely
murray.stokely@windriver.com

I liked your Musings about the security of open source operating systems, but one thing sort of bugged me. It always kind of irks me when I see "OpenBSD" singled out above the other BSDs for security.

Sure they did a stellar job a couple of years ago by doing an incredible amount of code review but the other BSDs have caught on now. Singling out OpenBSD is unfair for several reasons :

- FreeBSD supports a number of security-related features that the OpenBSD team just does not have the resources to develop, such as Mandatory Access Controls (DARPA-funded) and the lower-level mechanisms that make this work (See Robert Watson's papers at the last two Usenix conferences).
- The FreeBSD installation program prompts the user to decide if they would like to run inetd at all. Even if they choose yes, all services are turned off by default and they will have to be enabled in the installation program or turned on in the inetd.conf file (which contains nothing but commented out entries).
- FreeBSD offers the selection of "security profiles" during installation. These profiles can prevent the loading of kernel modules, changing of file flags, and various other potentially risky activities.
- FreeBSD has a very active security team doing independent code audits in addition to picking up all of the OpenBSD and NetBSD security changes. Indeed, our recent release of FreeBSD 4.4 was delayed for five days because of some security-related NetBSD changes that we wanted to include.
- There are many more eyes looking over the FreeBSD code than the OpenBSD code.

I certainly think that the OpenBSD guys deserve a lot of credit, and I don't want to argue about the relative merits of the different BSDs. I just think it's unfair to continue to single out OpenBSD as the "secure" BSD. When compared to FreeBSD, there are many security-related features that OpenBSD is lacking.

Murray – (BSD-enthusiast in general, FreeBSD developer in particular)

;login:

EDITORIAL STAFF

EDITORS:
Tina Darmohray *tmd@usenix.org*
Rob Kolstad *kolstad@usenix.org*

STANDARDS REPORT EDITOR:
David Blackwood *dave@usenix.org*

MANAGING EDITOR:
Alain Hénon *ah@usenix.org*

COPY EDITOR:
Steve Gilmartin

TYPESETTER:
Festina Lente

PROOFREADER:
Lesley Kay

MEMBERSHIP, PUBLICATIONS, AND CONFERENCES
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: 510 528 8649
FAX: 510 548 5738
Email: *office@usenix.org*
login@usenix.org
conference@usenix.org
WWW: *http://www.usenix.org*

needles in the craystack: when machines get sick

Epilogue: A Christmas Carol

by Mark Burgess

Mark is an associate professor at Oslo College and is the program chair for LISA 2001.

Mark.Burgess@iu.hio.no

> This is how one pictures the angel of history. . . .
> Where we perceive a chain of events, he sees one single catastrophe which keeps piling wreckage upon wreckage and hurls it in front of his feet.
> The angel would like to stay . . . and make whole what has been smashed.
> But a storm is blowing from Paradise . . . .
> This storm irresistibly propels him into the future to which his back is turned. . . . This storm is what we call progress.

Walter Benjamin, *Theses on the Philosophy of History, IX*.

It was the best of crimes, it was the worst of crimes. It was a crime of neglect and a crime of trespass. That night the old dinosaur changed his mind about many things. Even as the system folks flirted with the larger network community, old Scrooge would say: "Community meeting? Humbug!" tapping away on his spreadsheet. "Keep your head down. We have a job to do. Don't bother them, they won't bother us. Attention to work is the answer, not following every fad."

Every year, I think Christmas seemed to start earlier. It was only but the 2nd of November and the snow was already falling, piling into mounds. Of course, he was there in the office, as usual, going over the earnings of software sales. A client was visiting, trying to persuade Ebone to come to a USENIX conference, but Scrooge was ensnared by his spreadsheet, as usual.

She stood in front of the office window, looking out at the darkening weather. "You cannot truly appreciate the amount of noxious pollutants our machinery expires, until you've visited a country where the snow falls heavily," she said, turning to Scrooge for approval. "To see a snowy-white Winter Wonderland relentlessly transformed into disgusting black, roadside cement, just by car emissions, is one depressing sight which most of the world is spared."

"Nonsense, woman," Scrooge muttered. "It is every man's right to expend his resources as he sees fit. Cleaning up is simply a job which keeps someone in gainful employment."

"Gainful? Well, perhaps. But is it worthy employment? We could maybe avoid that particular challenge/response," she goaded, "if vehicles didn't ignore the world around them. How *does* one strike the balance between use and abuse?"

Scrooge sneered. "One simply bids the environment and its users to behave. End of the matter!" And that was Scrooge's philosophy, his answer to every question. Nose down and make thy fortune! Never mind the world at large, it is nothing more than a cumbersome distraction. "Are you going to turn at every little pinprick?" he would say.

Later, after she left, he resumed his sulking. The remainder of us were hoping to visit LISA, a conference on system administration, where we could revel in that environment which Scrooge dreaded so much, but he wasn't going to make it easy for anyone.

"I suppose you'll want all next week free!" he said.

"If convenient, sir."

"Well, it's not convenient. I fancy that if you spent as much time on our systems, as in your indulgence for staring into the space around us, we might achieve greater things."

But that night would come back to haunt him. Even as the client left, Scrooge was ignoring her warnings, ignoring everything around him. He believed he could just push the buttons and have his way. This time, the world was not going to obey him.

Later that night, after the other employees had escaped to their homes and families, Scrooge awoke before his terminal, alone in the office to the beeping of his mailbox. There was a message waiting for him. The icon on the screen had the form of a door-knocker. For a moment it seemed to blur and change into . . . no, humbug! He clicked on the knocker. A message appeared. It had the provocative subject "Pins and needles: watch your back!" He opened it irritably, imagining it to be crank advertising. Perhaps it was the late hour, and perhaps it was his sleepy imagination, but as the old man opened the message, it was not a window that appeared but something else entirely.

He stared at the screen. For a moment it had the appearance of the wife who had left him years ago for his stubbornness. Then the resemblance faded, and he was distracted by the clothing. It was rich and refined, but wrapped – no, almost mummified in chains. Heavy golden chains. Then, in surprise, he looked again. This time, it changed. In the dim light, there appeared a swelling in the air and a whine of straining hard disks. The screen melted away and something else took its place.

"What the Dickens!" he exclaimed.

"Indeed!" said a woman, standing before him in a blur of digital noise. A ghost? Surely, Cleopatra's ghost!

"I don't believe it!"

"Well, Scrooge," it said. "I am remotely here, just as surely as the zombie that Xwin passed. And I have come to warn you!"

"Warn me? Warn me of what? What is going on? I shall call security!"

"Security?" Her eyebrow lifted, ever so slightly. "What security? Well – you are going to find out soon enough, I fear." Then she focused. "You see my chains, Ebone? I constructed these chains, with my own hands! And now I carry them willingly! These are the chains of my past life. All my mistakes. It is my fate to wear them forever more. Now my spirit wanders the Net, with no other home. The only place where dreams and legends can be sustained. I am spread to the four corners, by my own foolishness."

"The chains . . ." Scrooge babbled.

"Yes, Markov chains. A record of my whole sorry past. I was blind to it then, you see. Even as they attacked me, the traitorous Roman barbarians. They had already seduced me, of course. My whole empire, infiltrated and by an attack on trust. Poisoned from the inside! You will make the same mistake, Ebone. I exist to warn those like you."

"Like me? What do you mean?" But even as he babbled, Cleopatra set about driving her little needles into him. Pinpricks of challenge, the seeds of uncertainty. She was building up for a different kind of attack: the infiltration of his attitude!

"Well, Scrooge," it said. "I am remotely here, just as surely as the zombie that Xwin passed. And I have come to warn you!"

It stared at him coldly – no welcome banner here. Just a cold prompt. "I am the horseman of entropy past," it croaked flatly

"Listen to me," she said. "I have come to introduce you to some friends of mine. Heed their warnings, Ebone! Don't end up like me. It's for your own security! Even as we speak, the attack is being mounted. Now I must go and give the others their say. Be good, Ebone. Look around you! There are whole worlds out there. A whole network out there . . . . You share a common space. It doesn't pay to ignore it! Believe in both the goodness and the danger of the environment and you can survive. It is not too late."

And with that she was gone, and the room seemed still and quiet, with only the beating of his heart, thumping away. At first he thought he was alone again, and began to gather his wits, even doubting what he had seen. What had she meant? An attack was being mounted? He was of a mind to dismiss the whole incident, surely a trick of the night – but the resemblance to his lost wife was astonishing. Cleopatra, of Egypt? Humbug!

Then, as he turned his head, his heart skipped a beat at a sudden braying in his left ear. A terrible apparition moved into the light, sitting astride a giant steed which clopped into view, and snorted with equine distain.

Upon it was a mere skeleton of a thing. A bare-bones interface. Not fleshed out with anything as user-friendly as a skin. It stared at him coldly – no welcome banner here. Just a cold prompt. "I am the horseman of entropy past," it croaked flatly.

"Another ghost!"

"Not a ghost, sir, but a projection! You might call me a model."

"A model you say? Not exactly The Lady Croft," Scrooge jibed. "And what is your message? Are you going to lecture me too?"

"Lecture you, sir? You misunderstand. A model is not mere theory. It is the embodiment of actuality! As a model, I am going to show you and then summarize the essence!"

Without further ado, the room began to dissolve around them. Suddenly they were standing in a small village. It had the appearance of Europe, he surmised, from the stone cottages. The horseman began to narrate like a bad movie, like the groaning of a great wheel. It seemed as relentless, as unstoppable as time itself.

"The story begins in a small village in the south of England," said the ghost. "An author, Mister Brunner, is writing a book called *Shockwave Rider*. It is a satirical vision of a future society with mobile phones, laptop computers, laser printers, and a world-spanning network!"

"Mobile phones? So you visited this . . . writer also? And showed him the future?"

The narration continued relentlessly, ignoring him. "In order to get jobs done users release 'worms' onto the network. Worms travel from computer to computer around the world, reading and writing information, both legally and illegally. The network is a busy place – a corporate war-field. This is the beginning. Later, a photocopying engineer will use the word 'virus'! A canny fellow. You must remember this, Scrooge." He placed a bony hand to the side of Scrooge's head. "Remember!"

"Humbug!" moaned Scrooge.

"A boiled confection in this time and place, I believe. I never cared much for those. Follow!"

The scene changed now. Even further back. Back to a time before civilization. It was a barbarous time. Humans killing humans, animals killing animals.

"See how they quarrel and fight?"

"Glad to see we stopped all that!" Scrooge parried.

"Ah, did we? Did we?" The horseman flashed his scythe, and boney teeth showed for a second. "My motto is this, sir: may the past come back to haunt you! It is never very far away. The Angel of History never quite has time you see. There is never enough time to fix things."

"But none of it is certain," he complained. "The past is the past. History does not necessarily repeat itself! We learn from our mistakes."

The horseman, stuffed an impromptu cigar into its hollow mouth. "Perhaps, perhaps not." Its hollow eye sockets seemed to narrow almost imperceptibly. "But then you gotta ask yourself a question . . . you feelin' lucky – punk?"

Then it departed, taking Scrooge back to the present, sitting in a pool of sweat. He sat in the office again. The snowy flakes were still falling outside. His hard breathing condensed on the screen in front of him. Somehow he could not deny its reality. the perspiration was real enough.

As the clock struck 18:00 EST, there was a chiming, a whirring of disks and a grinding of metal. The machines around him began to whir more slowly, as if made sluggish by the weight of a great burden. As he watched, it was not only the snow flakes that were falling in front of the real window: the numbers began to fall from their columns on his display window. He shrieked at the sight of wealth slipping away. Is this the attack? And a serpent entered through a back door.

"Is there no end to this?" Scrooge howled. The second of the apparitions laughed, and Scrooge winced at the sight of this beast. As it approached, machines stopped moving altogether, as if caught in a tar pit. The beast stank with a rotting stench of a thing full of bugs, and not quite wholesome.

"I am sickness," it announced.

"Don't tell me," muttered Scrooge weakly, "you're here to teach me drawing, smelling, and feinting in coils . . ."

"Sir," it hissed, "I am the worm of system present . . . . For you, this is not the best of times."

"Show me, dammit! Show me whatever it is you would, and be gone!"

It nodded, as only a worm can nod. "Even as we speak," it said, "it is happening. Look!"

The creature rolled and stared pointedly at the machines in the room around him. They had stopped altogether now, and each one was turning green and falling apart, as though infested by plague. There it was: a tiny needle-like infestation, piercing the stack of each machine. A small signal, on the scale of things, but as dangerous as a pestilence! The machines were being digested.

"Sir," it hissed, "I am the worm of system present . . . . For you, this is not the best of times."

"But we don't have the resources for that! We can't afford the time!"

"Pity though, because time can afford you. . ."

"This isn't happening," Scrooge groaned, seeing his wealth evaporate into a cloud of greenhouse gases.

"Well, of course, it's all just symbolism," smirked the apparition. "But that's the modern world for you. This will spread to the four corners if it is not stopped. But look at this."

The room faded and they were looking at Scrooge's greatest business rival's premises. One of their machines was dying, but the others were still alive. "They're surviving! This is the worst news yet!"

The worm prodded a man intently focused upon his terminal. "You see him? He's not ignoring anything. Security folks at work. This man spends his time thinking about the impact of the environment on his systems. It's not a one-off thing, Ebone – it's a continual process, and he is prepared for this. He is going to survive."

"But we don't have the resources for that! We can't afford the time!"

"Pity though, because time can afford you. Why do your machines get sick? Because I am here? Or because you have neglected them, and made them vulnerable? Because we provide an environment which is sub-optimal? Because there are conflicting, competing interests? Because human nature itself gives us attitudes and behavioral patterns which impinge directly on the machines we use? Watch this . . ." It pulled out a gun and they were suddenly in some kind of cowboy saloon. Scrooge was there, dressed as a cowboy. "I am going to shoot you," said the worm. "Protect yourself!"

Scrooge's manifestation leaped behind a column holding up the ceiling.

"You think so, eh?" The ghost winked and shot a column nearby. The bullet ricocheted off the column and hit him.

"Ouch! That's cheating!"

"Just using the environment to my advantage. If I hadn't, someone else would have."

"You shouldn't even be carrying a gun!"

"Ah, well, in this great country and so on, and so forth. Now I shall leave you. You have a lot to think about." It began to leave.

"Wait! Am I to believe that what is done is done? And that there is no purpose in crying over overflowed buffers? Well, that is what I knew all along, apparition! Progress will just bring on new catastrophes."

But the worm was burrowing into the Net, laughing as it went, and the illusion was fading.

Back in his office again, Ebone Scrooge (to his ghosts) surveyed the scene. I'm ruined! he thought. The machines were still silent, and now if his employees did not go to that LISA conference, they might never work again. Truly the worst of crimes. He had to admit that it was clever though. A computer-borne sickness – what an idea! But how to learn from the mistake?

He had little time to wallow in this pity. Soon enough, the third of the visitors was on its way. It appeared first at the end of a tunnel, approaching at the speed of illumination: not quite as fast as light, but comparable to the speed of enlightenment, a seemingly endless, tortuous rate of transfer. On the end of the tunnel was stamped "ACME

Information Superhighway (no warranty)." The third of the spirits was a metallic beast, but not quite like the machines Scrooge was used to.

"Don't tell me – you are the ghost of system future!"

The robot apparition did not speak. Instead it seemed to glow slightly, rotated a grasper as if beckoning and moved off, down the highway, towards the future, with Scrooge tagging along behind. This ghost of the future was a curious thing, artificial looking, but its posture – its whole disposition – was oddly human.

They emerged into a shambles. Not quite the future Scrooge had imagined when he invested in high-tech stocks. An old man sat in a dingy office, totting up numbers on a pocket calculator. The Angel of Computer History scarcely acknowledged their presence, if he could even see them at all.

"Ah, so you're back," it mumbled eventually, as if talking to an imaginary friend. "And you've brought another. Do you know me? Perhaps not." He shook his head. "I was once assigned to keep computing systems whole, you know, to repair them," he mumbled. "But it is hopeless. It all went wrong, you see. When the worms came."

Scrooge barely mouthed a question when it was answered. "Why am I doing this by hand? The computers are up there on the hill. No one has computers themselves anymore. They're all locked up, out of harm's way. Too much trouble. Now it all costs too much. The computers are just for the elite." The man tapped the side of his head. "No sooner built than destroyed. All smashed."

Scrooge looked to his guide for explanation, but the old man simply continued as if he knew the drill.

"The breakdown of order on the network rendered it useless. The level of noise was finally so high that no meaningful signal could be safely transmitted. It turned out to be just work for nothing. All wasted. That's what happens when those who *have* pay no attention to those who *haven't* – or perhaps wouldn't . . ." He chuckled. "So we're back to paper again." The old man, or Angel, took a breath and articulated more powerfully. "By building distributed systems, they increased the parallelism in computer systems, but thereby also increased the contact surface with the external world – an easier target. As they ignored the signals from the network environment, as they ignored the time-given laws of community, the attacks increased. So now, you see, we are back to the beginning. Any system can develop a virus, you know. Viruses emerge from the very systems themselves! Any protocol can be attacked. Any influence, however small, can be amplified by the right conditions into a potential problem. The chains of cause and effect are both devious and intricate. But the worst ones won. The warfare reached its peak. Sometimes political, sometimes just animals flexing their muscles. It was the return of the dinosaurs. Dog eat dog. Did I hear you say that history does not repeat itself?"

As he faded somewhat from view, he was still laughing. A slow, rueful laugh, not a triumphant one. Scrooge shuddered. This rambling madman was not a future he wanted to revisit. No computers? All because of the worms?

He found himself in a school, perhaps. No, it was a class group on a visit. There was a guide leading them around a factory of some kind. The children brought out their slates to make notes. One of the children raised a hand. "Miss, my slate's not feeling so great." It was duller than usual, and its login face showed visible distress. The tempera-

"The breakdown of order on the network rendered it useless. The level of noise was finally so high that no meaningful signal could be safely transmitted. It turned out to be just work for nothing. All wasted."

1. A computer may not cause a user harm, or through inaction allow a user to be harmed.
2. A computer must obey instructions given to it, except where this would conflict with the first law.
3. A computer should protect its own integrity, as long as this does not conflict with the first two laws.

ture scale on the left of the screen showed that it was running a temperature, fighting some illness. It had a dour expression.

"You'd better leave it to rest for a few minutes," said the teacher. "You can borrow another, if you're kind to it. It won't be used to you, so be nice!"

The child took a new slate and wandered off, knowing that his own would right itself shortly. They were looking at the computers of a different future, Scrooge presumed, though they didn't look much like computers. They were more organic – not in the biological sense, but in the sense of being like an organism. Even though he could not see what programs they were running, the shine and outward demeanor of the machines seemed to be visible. The materials seemed to change and project this character. As the children used them, they responded to one another. They were more like robots than computers, but not mobile robots, and they interacted, not merely at the level of commands, but in a more socially savvy way. It seemed to be intuitively obvious that the machines were feeling good, bad, or simply stressed. Users avoided the ones which seemed ill, allowing them to recover by themselves. Ingenious!

They came upon a machine which did not look sick, but it did not respond. "What happened to this one?" someone asked.

The guide relayed, "We build machines which have feelings, so that they can react to protect themselves from all the complex things happening to them. They need to know good from bad, right from wrong to do that. But this machine was built with too many feelings," said the guide. "It eventually developed its own sense of right and wrong and ended up in a quandary. It decided it didn't like doing what we built it for. Now no one dares turn it off, because of the computer rights activists. Nor can we do anything to change its mind. Basically, it's a junkie, locked in its own world. We call it the e-dopamine syndrome. When you let the machine adapt, change the playing field from being flat, you'd better do it right. We went a little too far. We made a living thing, and all we wanted was a machine."

"You are telling me that computers get sick here? How?!" Scrooge demanded.

"When didn't computers get sick? Why does anything get sick? Stuff gets mixed up. Shit happens," quipped the guide, then returned his attentions to the children. "There are many sicknesses in systems. Technophobia is a sickness of society. Warfare is a sickness of society. Even the free-market economy has led to sicknesses, though at least it produced antibodies too. Our feeling machines tend to spread the load by turning large-scale conflict or warfare into small emotional bickerings. It spreads the chaos – I mean the entropy. It allows the systems to let off steam."

There was a plaque by the machine he was talking about now, inscribed with the three laws of interactive computers.

1. A computer may not cause a user harm, or through inaction allow a user to be harmed.
2. A computer must obey instructions given to it, except where this would conflict with the first law.
3. A computer should protect its own integrity, as long as this does not conflict with the first two laws.

This is nothing but magic, thought Scrooge. This will never happen.

The echo of the Angel of History reached his ears: "Clarke's law: any sufficiently advanced technology is indistinguishable from magic." Yes, yes, he thought, but that doesn't make it possible! The Angel, as if reading his mind, parried: "A distinguished scientist who says it's possible . . . is probably right. A distinguished scientist who says something is impossible . . . is probably wrong." Clarke's other law.

"Humbug! You are just telling me that I should be aware of my competitors. I already knew that!" They seemed to fade out of this future as they talked. As the children receded into the distance, the Angel rejoined them, picking the pins off a chip, one by one, like the petals of a daisy. "Big kernel, small kernel, big kernel, small kernel. He believes me, he believes me not . . ."

"Well," continued the Angel, rematerializing more tangibly, "discussions on competition focus too much on winning. Winning implies a certain finality, an end to conflict, that would have us pack up our systems and leave after every altercation. This is naive. The conflict goes on. It never ends. We must be concerned with holding the forces of evil in abeyance. More than that would be overly ambitious. Our strategies need to maintain stalemate or minimize the damage. These altercations should not be the focus – we are concerned with the larger goals of producing work, the acquisition of assets. There is no time to repair the little stuff. I realize that now. It's acceptable loss."

"It's a catastrophe. All my systems!"

"Catastrophe in an ecosystem often clears the way for change. Forest fires clear old wood. Hurricanes throw a random die into the balance of power. Ice-ages, dinosaur killers . . . . Our worm friend has merely cleared the way for your understanding." The robot of system future whirred and touched Scrooge on the shoulder. The Angel looked at his timepiece: a small wind sail, attached to his sleeve. "You should be returning. It is time."

Scrooge nodded. He finally understood. He had been wrong to assume that his push-button mentality was the answer. One cannot simply decide to resist the onslaught of environment. Even the Angel had realized that. Even with a large umbrella, you'll get wet in the rain. Yes, he understood. It was about sharing with neighbors, and watching out for them! I shall call it the principle of communities! he thought. When one shares a common space with one's neighbors, interaction is inevitable. Best to make sure that those interactions are pleasant ones. He would recite it to himself on the way back to the present. Now, how to get to that conference? What was it called again?

And that is the story of how Ebone Scrooge learned the true meaning of Christmas.

"Big kernel, small kernel, big kernel, small kernel. He believes me, he believes me not . . ."

searching through your files with glimpse

by Bob Gray

Bob Gray is co-founder of Boulder Labs, a digital video company. Designing architectures for performance has been his focus ever since he built an image processor system on UNIX in the late 1970s. He has a Ph.D. in computer science from the University of Colorado.

bob@cs.colorado.edu

You have hundreds of megabytes of emails, FAQs, documents, and source code. You need to find something that you only vaguely remember. What are you going to do? You could start looking with an editor, you could try grep, but there is a better way.

Recently someone asked me about resisting poison ivy while hiking. I knew I had an email or FAQ about the topic, but it had been years since I had saved the information. In two seconds, I located the article with this command:

```
% glimpse -W 'poison;ivy'
```

In contrast, a recursive version of egrep required 300 seconds to search my 11,000 files totaling 250MB. Further, the command

```
% find . -print | xargs egrep 'poison|ivy'
```

yields dozens of inappropriate matches (including binary files) because it matches lines containing either "poison" or "ivy", whereas the glimpse -W option requires that both words be present in the same file.

Glimpse is an indexing and query system that allows you to search through files very quickly. Glimpse has a lot of overlapping capabilities with grep, but they each have their own sweet spots. In this article, I'll start with a few examples, then I'll provide some background. We'll look at the features of this tool and show its performance. By the end, you'll have enough information for deciding whether to add glimpse to your repertoire.

Glimpse: Practical Examples

My email folders are reasonably tidy – I delete unneeded messages, yet my email still consumes more than 25MB in over 4000 messages. My email client, exmh, presorts new mail into a hierarchy of files rooted in the directory $HOME/Mail. Often, I need to retrieve old messages that I only vaguely remember. Using glimpse, it's easy to find the desired message in the $HOME/mail tree:

```
% glimpse -F Mail 'master;boot;record'
```

If necessary, the search can be improved with the case insensitive option, -i, the complete word option, -w, and/or the file as a record option, -W (more details below). The authors of glimpse even suggest that you alias glimpse with 'glimpse -i -w' because it's generally most useful.

At the University of Colorado (see "Teaching Operating Systems with Source Code UNIX")[1], I insist that the students load glimpse to aid in working with a large body of code. Tracing through function calls or variables is easy once an index exists. One of my goals for this class is demonstrating how to get comfortable with a large, unfamiliar code base. Tools such as glimpse and editor tags are essential (man etags).

Glimpse History

Glimpse was developed by Udi Manber and Burra Gopal of the University of Arizona and Sun Wu of the National Chung-Cheng University, Taiwan. They published "GLIMPSE: A Tool to Search through Entire File Systems" in the 1994 Winter USENIX Proceedings. The paper is on the USENIX Web site.[2] Much of glimpse is based on their earlier work with agrep (see "AGREP - A Fast Approximate Pattern-Matching Tool," published in the 1992 USENIX Proceedings).[3]

Glimpse has continued to evolve over the years, and there now is a cooperative development organization (see *http://webglimpse.org/*) to advance this software and its derivatives. To support the effort, they collect a license fee when glimpse is used commercially.

Although not a part of standard UNIX[4] distributions, glimpse is freely available. There are Linux RPMs,[5] precompiled binaries, and an entry in the FreeBSD ports tree. Glimpse 3.6, available from,[6] can be used without licenses. Glimpse 4.12.6[7] is free for noncommercial use, but commercial use requires a license. Glimpse source code is available from many locations. An "archie" search will enumerate source code sites. Try: *http://elfikom.physik.uni-oldenburg.de/Docs/net-serv/*archie-*gate.html* (use the keyword "glimpse-3").

Glimpse Features

Glimpse, like grep, is a UNIX searching tool that helps you find content in files. Whereas grep finds patterns in one or more files by on-the-spot examination, glimpse instead consults a pre-built index to perform the query. The advantage is speed – files comprising hundreds of megabytes can be searched in seconds. The disadvantage is the extra space and time required to compute the index. Assuming a hierarchy of ASCII files, the index requires an additional 2–3% disk space, or for maximum performance, 20–30%. The time to compute the index is on the order of the time it takes to grep through the same files. But I keep a fresh index ready for searching with a crontab entry.

```
53 4 * * * /usr/local/bin/glimpseindex . >/dev/null
```

builds my home directory (".") glimpse index every morning at 4:53 a.m. and stores it under $HOME. Alternate indexes can be built for any hierarchy and stored in an arbitrary directory using the -H option. Let's run through a few examples to show various glimpse features:

```
% glimpse windsurfing
```

will match lines that contain the target word. A '-i' will make the search case insensitive.

```
% glimpse 'Arizona desert;windsurfing'
```

will find all lines that contain both "Arizona desert" and "windsurfing".

```
% glimpse -W 'license;hash;expired;features'
```

requires that all four words exist somewhere in the file. For those files, glimpse will output the lines that contain any of the words.

```
% glimpse -w ivy
```

requires complete word match; "divy" and "bivy" won't match.

Glimpse, like grep, is a UNIX searching tool that helps you find content in files.

Glimpse has flexibility on the definition of a record.

```
% glimpse -F '\.c$' union
```

searches for the word "union" in "C" files. The -F option limits the search to those files whose name matches the given parameter: in this case, files ending with the C file suffix .c (for example, kern/vfs_bio.c and vm/vnode_pager.c). The -F option allows a case-insensitive flag, so -F '-i faq' would look in file names containing "faq", "Faq", etc. And -F '-v \.c$' would conduct the search in anything BUT C files.

```
% glimpse -2 pneumonic
```

will find all occurrences of "pneumonic" allowing two spelling errors. That would include "mnemonic", "pneumonia", and "newmonics". This feature is part of agrep, where an integer between 1 and 8 specifies the maximum number of errors permitted in finding the approximate match (the default is zero). Generally, each insertion, deletion, or substitution counts as one error. Also from agrep is the Boolean matching concept illustrated in the next two examples.

```
% glimpse '{political,computer};science'
```

will match lines with any of these strings: "political science", "computer science", or "science of computers".

```
% glimpse -W 'fame;~glory'
```

will output all lines containing "fame" in all files that contain "fame" but do not contain "glory".

```
glimpse -i -F 'faq$' -d '$$' 'master;boot;record'
```

Glimpse has flexibility on the definition of a record. The -d option allows you to override the default record delimiter, '$', that is, a line is a record. In the example, -d '$$' defines paragraphs as records, so in any file name ending with 'faq', it will find occurrences of 'master;boot;record' all in the same paragraph. For searching in files containing email the option -d '^From ' defines records as entire email messages.

I've highlighted the features of glimpse that I've found most useful over the years. Read the manual page to see how glimpse can best help you.

Glimpse Performance

This section gives some time and space requirements of glimpse. I'll measure performance on the freely available FreeBSD 4.4 kernel sources of September 2001 so that my experiments can be repeated by the readers. This is a rather small sample to index, but it is still useful and realistic for those needing to deal with kernel source code. I'll use a modest 200 MHz, 32MB PC with a SCSI disk that's a few years old. Let's characterize the body of source code.

```
# cd /usr/src/sys
# du .
574 ./alpha/alpha

...
248 ./ufs/ufs
646 ./ufs
536 ./vm
-----
48083 .
```

```
# find . -type f -print | xargs wc
...
      1018       3552      25067 ./vm/vnode_pager.c
        61        408       2777 ./vm/vnode_pager.h
    512603    1869237   14093776 total

# find . -type f -print | wc
     3472     3472    80303
```

There are 3,472 source code files taking up 48MB of disk space. In total, the kernel consists of 512,603 lines, 1,869,237 words, and 14,093,776 characters. We'll measure how long grep takes to make a typical search in this code base and then look at the elapsed time for glimpse, assuming the index exists. (For measuring time, I'll use the built-in shell command time and report only the elapsed-time component).

```
# cd /usr/src/sys
# find . -type f -print | xargs grep vm_pageout_deficit
./kern/vfs_bio.c: vm_pageout_deficit += ...
...
./vm/vm_pageout.h:extern int vm_pageout_deficit
elapsed time: 39 seconds

# glimpse -H . vm_pageout_deficit
kern/vfs_bio.c: vm_pageout_deficit += ...
...
vm/vm_pageout.h: extern int vm_pageout_deficit;
elapsed time: 0.5 seconds
```

Clearly, glimpse enables much faster searching. (The -H option tells glimpse to consult the index in the current directory.) If you own a 1 GHz PC, don't assume you could search five times faster than with a 200 MHz PC. Realize that grep is mostly an I/O-bound process because you have to read 3,472 files to conduct the search. Let's look at the cost of building the index. As with grep, glimpseindex is also I/O bound.

```
# glimpseindex -H . .

Indexing "/usr/src/sys" ...

Size of files being indexed = 45988952 B, Total #of files = 3457 ...

-rw------- 1 root    117885    Sep 28 08:55 .glimpse_filenames
-rw------- 1 root     13828    Sep 28 08:55 .glimpse_filenames_index
-rw------- 1 root   2563925    Sep 28 08:55 .glimpse_index
-rw------- 1 root       417    Sep 28 08:55 .glimpse_messages
-rw------- 1 root       880    Sep 28 08:55 .glimpse_partitions
-rw------- 1 root     12341    Sep 28 08:55 .glimpse_statistics
elapsed time: 94 seconds
```

Glimpseindex builds an index of the tree rooted at "." and, with "-H .", stores it in the current directory. Remember, you don't need to run glimpseindex very often, so the 94 seconds can support a lot of cheap glimpse searches. The index size is almost 3MB to index 48MB of data, or 6%. The glimpse authors recommend that most casual users create the smallest index by specifying the -o option. And where searching speed is paramount, build a larger index with the -b option.

The astute reader may notice that glimpseindex reports: "Total #of files = 3457" but "find . -type f -print | wc" reports 3,472. That's because there are a couple of binary files in the hierarchy that glimpseindex skips. It also makes an effort to identify and skip

Clearly, glimpse enables much faster searching.

REFERENCES

1. *http://boulderlabs.com/12.teaching*.

2. *http://www.usenix.org/publications/library/proceedings/sf94/full_papers/manber.glimpse*.

3. *http://www.usenix.org/publications/library/proceedings/wu.pdf*.

4. As always, I loosely use the term "UNIX" to mean UNIX-like systems, including Linux, {Free, Open, Net}BSD, Solaris, etc.

5. *ftp://linux1.fnal.gov/linux/611/SRPMS//glimpse-4.1-4.src.rpm*.

6. *ftp://ftp.cs.tu-berlin.de/pub/linux/Mirrors/sunsite.unc.edu/utils/text/glimpse-3.6.src.tgz*.

7. *ftp://ftp.polito.it/pub/tools/unix/harvest/glimpse-4.12.6.tar.gz*.

8. *http://swexpert.com/* (click on UNIX Basics).

9. *http://webglimpse.org/*.

10. *http://www.tardis.ed.ac.uk/harvest/* and *http://www.si.uniovi.es/mirror/harvest-net/*.

other non-ASCII files such as compressed, uuencoded, and postscript files. You can customize the skipped files with a .glimpse_exclude file.

On the other hand, sometimes you want to index what is kept in compressed files. Using a .glimpse_include file, you can arrange for glimpseindex to examine otherwise ignored files. The .glimpse_filters file allows you to specify a program to explode the coded files so that glimpseindex has something to work with. For example, if .glimpse_filters includes the line

```
*.Z uncompress <
```

then any file ending in .Z is uncompressed before glimpseindex sees it. The file itself is not changed (i.e., it stays compressed).

Miscellaneous

Three quick side notes: first, my colleague Peter Collinson just wrote an excellent tutorial, "Grep Is Fundamental," in the September 2001 Server/Workstation Expert.[8]

Second, Webglimpse is a by-product of glimpse for indexing Web sites.[9] Its predecessor, Harvest, provides some interesting history.[10]

Third, sometimes it's not the content you want to search but just the file names. I've mentioned that I automatically create a daily glimpse index. I also create a FIND file:

```
find $HOME -print > $HOME/.DOT/.FIND
```

I have a shell script, ef, that consults this list to help me locate file names:

```
% ef backpacking
/usr/people/bob/BACKPACKING
/usr/people/bob/BACKPACKING/REPAIRS
/usr/people/bob/BACKPACKING/food
/usr/people/bob/BACKPACKING/BP-LIST
```

The first argument to the script is the string I'm looking for. Successive arguments are filters to eliminate noise. Here is the essence of the ef script:

```
lst=$HOME/.DOT/.FIND
case $# in
1) egrep -i $1 $lst ;;
2) egrep -i $1 $lst | egrep -v $2 ;;
3) egrep -i $1 $lst | egrep -v $2 | egrep -v $3 ;;
4) egrep -i $1 $lst | egrep -v $2 | egrep -v $3 | egrep -v $4 ;;
```

To find any file name, give it as the first argument. If you "hit-the-jackpot," start adding filter arguments until the list shows just what you want. Try it and it will become clear.

Thanks to reviewers Dave Clements, Tom Poindexter, and Steve Gaede.

some new numeric programming features in c9x

We've been looking at some of the changes in the C9X revision of the C language standard. In this column, we'll consider several new features in the numeric programming area.

by Glen McCluskey

Glen McCluskey is a consultant with 20 years of experience and has focused on programming languages since 1988. He specializes in Java and C++ performance, testing, and technical documentation areas.

glenm@glenmccl.com

New Integer Types

Suppose that you're doing some C programming and you need to work with 32-bit integers. Which C type should you use for this? You could say:

```
int x;
```

but there's no guarantee that int is 32 bits. With old PCs or embedded applications, it might be 16, and on a high-end workstation or a supercomputer, 64 or 128. A standard way of dealing with this problem is to define a header file with typedefs in it, like this:

```
typedef long INT32;
```

and then use INT32 everywhere.

In C9X, this mechanism has been formalized through the stdint.h header file. Here's a simple example:

```
#include <stdint.h>
#include <stdio.h>
#define N 100

int32_t vector[N];

int main()
{
    for (int i = 0; i < N; i++)
        vector[i] = 0x7fffffff;

    printf("vector[59] = %d\n", vector[59]);

    return 0;
}
```

int32_t is a signed integer type of exactly 32 bits. We can go further in using stdint.h types in this example, and come up with the following:

```
#include <stdint.h>
#include <stdio.h>

#define N 100

int32_t vector[N];

int main()
{
    for (uint_fast16_t i = 0; i < N; i++)
        vector[i] = INT32_MAX;

    printf("vector[59] = %d\n", vector[59]);

    return 0;
}
```

uint_fast16 is another typedef, specifying an unsigned integer type of at least 16 bits, that is the fastest for your local hardware. The idea is that you know you need at least a 16-bit unsigned type, and you let the system pick the best one for you.

INT32_MAX is a macro that gives the maximum value for a signed 32-bit type.

Another type in stdint.h is intptr_t, a type that is guaranteed to hold a void* pointer, such that you can convert a void* to intptr_t and back, without any loss of information. Here's an example of how you would use intptr_t:

```
#include <stdint.h>
#include <stdio.h>

int main()
{
    void* p1 = (void*)0x12345678;

    intptr_t saveptr = (intptr_t)p1;

    void* p2 = (void*)saveptr;
    printf("p2 = %lx\n", p2);

    return 0;
}
```

Working with Integer Types

There's another header file, inttypes.h, that provides some utilities for working with the integer types described above. To illustrate these utilities, here's an example that shows how you can use the intmax_t type, a type that specifies the maximum-size integer available on your machine:

```
#include <inttypes.h>
#include <stdio.h>

int main()
{
    intmax_t val = INTMAX_MAX;

    printf("val = %" PRIdMAX "\n", val);

    return 0;
}
```

inttypes.h includes stdint.h, so you don't have to explicitly include it.

The program defines a variable of type intmax_t and sets it to the maximum value. The value is then printed. PRIdMAX is a string defined in inttypes.h that specifies the appropriate printf format for integer ("d") variables of maximum width ("MAX"). On my machine, given that intmax_t is long long, this format is "lld". The three juxtaposed strings in the printf statement are concatenated. Note that the standard requires that intmax_t be at least 64 bits.

Given this ability to specify the printf format in a portable way, we can go back and modify a previous example a little more:

```
#include <inttypes.h>
#include <stdio.h>

#define N 100
```

```
int32_t vector[N];

int main()
{
    for (uint_fast16_t i = 0; i < N; i++)
        vector[i] = INT32_MAX;

    printf("vector[59] = %" PRId32 "\n", vector[59]);

    return 0;
}
```

PRId32 is used to format 32-bit decimal integers.

Another utility offered in inttypes.h is one that lets you do integer division with intmax_t values, obtaining both the quotient and remainder in one operation. Here's an example:

```
#include <inttypes.h>
#include <stdio.h>

int main()
{
    intmax_t num = 987654321;
    intmax_t denom = 123456789;
    imaxdiv_t res = imaxdiv(num, denom);

    printf("quotient = %" PRIdMAX "\n", res.quot);
    printf("remainder = %" PRIdMAX "\n", res.rem);

    return 0;
}
```

There's also a function for taking the absolute value.

inttypes.h also specifies a group of functions that you use to convert strings to intmax_t values. Here's a demo program that shows one of these functions:

```
#include <ctype.h>
#include <inttypes.h>
#include <stdio.h>

int main()

{

    char* input = "1,  123456789123456789,  37,-987654321  ,0,59";
    char* currptr = input;
    char* endptr;

    for (;;) {
        while (*currptr &&
        !(isdigit(*currptr) || *currptr == '-'))
            currptr++;

         if (!*currptr)
            break;
        intmax_t val = strtoimax(currptr, &endptr, 10);
        printf("val = %" PRIdMAX "\n", val); currptr = endptr;
    }

    return 0;
}
```

strtoimax is a function that parses an input string, converts it to an intmax_t value, and returns an updated string pointer so that you can step through the string.

You can specify the number base to strtoimax, as this example shows:

```
#include <inttypes.h>
#include <stdio.h>

int main()
{
    intmax_t val = strtoimax("1111111111111111111",  0,  2);

    printf("val = %" PRIdMAX "\n", val);

    return 0;
}
```

The output from the program is 1048575.

Type Generic Math

Standard math functions like cos typically accept an argument of type double. There are times when you'd like to operate on floats, for space or speed reasons, or on long doubles, to get extra precision. And you might want to use complex types as well, given that C9X supports complex arithmetic.

There are new functions in the standard for working with float and long double and complex types. For example:

cosf	cosine function for float
cosl	cosine function for long double
ccosl	cosine function for complex long double

In addition to these functions, there is a facility defined in tgmath.h, that lets you use a single function (cos) for all these cases, with the "right thing" automatically done for you by the compiler and library; that is, the right function is called based on the argument type. Here's an example that illustrates how this works:

```
#include <stdio.h>
#include <tgmath.h>

int main()
{
    float f = 0.123456;
    double d = 0.234567;
    long double ld = 0.345678;
    complex long double cld = 0.456789;

    if (cos(f) != cos(f))
        printf("cosf error\n");

    if (cos(d) != cos(d))
        printf("cos error\n");

    if (cosl(ld) != cos(ld))
        printf("cosl error\n");

    if (ccosl(cld) != cos(cld))
        printf("ccosl error\n");

    return 0;
}
```

Another way you can look at what's happening with generic dispatching is shown in this demo:

```
#include <complex.h>
#include <stdio.h>
#include <tgmath.h>

int main()
{
    printf("%d\n", sizeof(cos((float)0)));
    printf("%d\n", sizeof(cos((double)0)));
    printf("%d\n", sizeof(cos((long double)0)));

    printf("%d\n", sizeof(cos((complex float)0)));
    printf("%d\n", sizeof(cos((complex double)0)));
    printf("%d\n", sizeof(cos((complex long double)0)));

    return 0;
}
```

On my machine, the sizes of the results of the cos() calls range from 4 (float) to 24 (complex long double), indicating that different versions of the cos function are indeed being called.

Here's another example, using sqrt:

```
#include <complex.h>
#include <stdio.h>
#include <tgmath.h>

int main()
{
    complex double d = 37.0 + 47.0 * I;
    complex double s1 = sqrt(d);
    complex double s2 = csqrt(d);

    printf("%g %g\n", creal(s1), cimag(s1));
    printf("%g %g\n", creal(s2), cimag(s2));

    return 0;
}
```

When you run this program, the result is:

```
6.9576    3.3776

6.9576    3.3776
```

The type generic feature is similar to C++ function overloading and allows for portable code to be written. You can use all of the features we've described above in this way, to write efficient code that's easy to move from one machine and compiler to another.

a quick introduction to database systems

Richard Leyton

Richard Leyton is a senior consultant at Paremus Limited, a newly established technology consultancy company (*http://www.paremus.com*). He has over 10 years experience with UNIX and has worked with various database system installations in and around finance and the dot-com industry.

richard@leyton.org

Introduction

One important area of computing systems management is often overlooked by system administrators but accounts for some of the biggest, most complex, and frequently most important systems for which we are responsible. That area is databases, and they're overlooked because they're perceived by many to be boring, concerned with old technology, unreliable, and the cause of many headaches. To top it all, too often they don't really do very much that's noticeable or interesting.

It's my belief that none of these perceptions have any validity; databases are an interesting, challenging, and evolving area of technology, which, if implemented and supported well, can bring a perceptible benefit to the system administrator, the system itself, and, of course, the users of the systems and of the institution.

What Is a Database?

In its very simplest form, a database can be viewed as a "repository for data." Tautological as it sounds, this repository is tasked with maintaining and presenting the data in a consistent and efficient fashion to the applications, and the users of such applications, that use it. It is these factors which complicate the matter.

Before databases appeared as a separate technology, data was stored in a variety of ways, often proprietary and specific to the implementation in question. Data couldn't be shared and couldn't be utilized outside of the application in which it resided. This clearly proved problematic – a company had the data, but couldn't do more with it — as new requirements came about.

Databases evolved to take responsibility for the data away from the application, and, most importantly, enable it to be shared. As applications grew and new applications appeared, a single data repository evolved, a repository that all applications could access (in an agreed format and model, of course).

Of the many forms possible, today's databases are usually "relational databases." This is not the only variety but has gained ascendancy because it is simple and effective. Older models include the "hierarchical" and "network" (N.B., not like the Internet) models, which can still be found in legacy mainframe environments. These models lost favor because they focused on storage issues rather than data issues. Newer and increasingly popular data models include object-oriented and object-relational, which can in certain circumstances map nicely to object-oriented systems.

But relational databases continue to form the bulk of database systems and are the focus of most books on database design and implementations. Relational databases became popular because they stripped away the machine-specific storage mechanics of the older models so developers no longer needed to worry about how the data was stored and how to retrieve it; they could focus on the data itself and concentrate on building functionality-rich applications.

Oracle (producer of one of the first commercial relational database implementations) was formed out of the research work undertaken at IBM on their System/R research work. The rest, as they say, is history. Now IBM, Sybase, Computer Associates, and various others have established very mature, stable products (though they are not market leaders). And new companies are entering the market all of the time: Clustra, RedHat,

Versant, and the GNU project are all producing database systems that offer something new and innovative, keeping the established players on their toes.

What Must a Database Product Provide?

- Consistency: It must ensure that the data itself is not only consistently stored but can be retrieved efficiently. This is even more critical when changes to the data occur without warning.
- Concurrency: It must enable multiple users and systems to all retrieve the data at the same time and to do so logically and consistently. Concurrency problems will be familiar to many readers, but in a database environment, concurrency is further complicated by the necessity to undo changes made in certain circumstances (e.g., deadlocks and aborted transactions).
- Performance: Users will be very demanding if faced with long response times. Scaling to cope with large numbers of users, all with demands on the resources, can become complex (but it's not rocket science). The database administrator can help by reviewing the access strategies to the data (indexes, caching and compute resources). Sometimes, a very simple change to some or all of these components can significantly improve performance (and sometimes decrease it elsewhere).
- Standard adherence: Most people have heard of SQL (Structured Query Language). It was envisaged by the original researchers at IBM as a query language designed specifically for the relational model to enable programmers to specify how the data should be extracted from the database in an easy way that is independent of the programming language being used. Most databases support the ANSI-ratified SQL92. Additionally, connectivity standards are required. Two of them – ODBC and JDBC – provide common APIs to the database, which gives developers a greater degree of flexibility over which underlying platform they use.
- Security: A database that provides access to any data for any user and also allows them to change it is not really suitable to many business applications. Database systems solve this through access permissions (much like files at the operating-system level) and specific database mechanisms such as triggers.
- Reliability: Of course, databases must keep their stored data intact. Additionally, coping well when things go awry is often a good indicator of the strength of a system administrator and the strength of a database system. A database must, if set up properly, be able to recover to a known consistent point. The use of write-ahead logs (transaction logs) facilitates this but can introduce performance bottlenecks. Needless to say, after repairing a faulty disk array, the very very last thing an administrator wants to deal with is a corrupted or unusable database.

Beyond the Basics

Once the idea of databases was established, databases could store and retrieve data efficiently, effectively, and reliably. Then vendors began to add features to enhance this basic functionality and give them a competitive edge. Some of the extensions have included the following.

Programming Languages to Manage the Data

Oracle has PL/SQL; Sybase/Microsoft have T-SQL. These languages go beyond the de facto standard "SQL" and add functionality (iterative loops, variables, procedures and

A database that provides access to any data for any user and also allows them to change it is not really suitable to many business applications.

Data is really only useful if it has some meaning.

mathematical functions) you'd normally find in more commonly known programming languages. This helps users manage data effectively but also reduces portability.

Maintenance of Data Integrity

Data is really only useful if it has some meaning (i.e., data in the "employee" table that is only employee information and not corrupted by, say, supermarket prices). When data is inserted, deleted, or modified in the database, implicit meaning can be (or might need to be) associated with that data. By using mechanisms known as "triggers" (code that is executed on such events), databases can maintain, introduce, or enforce the meaning. For example, when adding an employee to a database, checks are made to ensure that their social security number is stored and that their manager is defined.

Connectivity

A client application must be able to communicate effectively with the database. Vendors often produce native drivers/libraries for client programs in order to enable efficient connections and queries. However, in this time of open standards, several new bridging and connectivity standards enable programmers to program independently of the actual underlying database: ODBC (the Microsoft-instigated Open Database Connectivity), JDBC (Java Database Connectivity), and Roguewave's DBTools are the three most widely known.

Unfortunately, database independence too often comes at a cost, as it often becomes difficult to avoid using a vendor's features as a quick solution for a complex problem. This can place a greater burden on the application developer as the client or application server might need to undertake more work. Furthermore, performance can also decline since only SQL92 standard queries can be used. The matching (impedance, if you will) between a set of standard function calls and the vendor's calls (especially in older database client libraries) can incur a client-side penalty.

Redundancy/Reliability/Recovery

Over the last few years, highly available systems have been demanded, with downtime of, at worst, minutes per month (five minutes per month is 99.99% reliability) rather than hours per month (99.7% reliability is just two hours per month of downtime).

Database vendors have been somewhat slow to recognize this, but products and solutions are now widely available. Many of them take the approach that high availability needs to be offered in conjunction with operating system vendor cluster/high-availability solutions. Others take the approach that operating systems can't be trusted in this regard and implement a distributed redundant approach themselves as an integral part of the product.

Recovery of a failed system (or resorting to a known-safe point in time) is crucial, but backups of huge systems can take a correspondingly huge amount of time, even with the best backup system in the world. Incremental dumps are vital too. Being able to restore a system to a particular point in time is important, especially when dealing with time-sensitive data or situations. By dumping out the transaction/activity logs, many database vendors have been able to offer acceptable backup solutions to a very fine level of granularity.

Getting Started

It would be foolish to assume that a short article like this can cover the entire subject area of database systems. But hopefully it has presented some of the basics. There are, of course, plenty of books that serve as good, comprehensive introductions to databases, which the interested reader might wish to consider.

C.J. Date's *An Introduction to Database Systems* is widely considered to be the best all around, in-depth book.

Theory and Practice of Relational Databases, by Stefan Stanczyk, Bob Champion, and Richard Leyton ably initiates the reader into both the theory and practice issues of databases. For more information, visit *http://www.theorypractice.org*.

There are plenty of resources online, too. Here are two of the best:

http://directory.google.com/Top/Computers/Software/Databases/
http://uk.dir.yahoo.com/Computers_and_Internet/Software/Databases/

Next Up

In upcoming issues, some of the following areas will be covered in more depth:

- Recent developments and innovations in database technology
- Open source databases vs. closed source databases
- Performance tuning database installations
- Improving reliability of database installations
- Integrating databases in corporate environments

References

The paper that started the relational model: *http://www.acm.org/classics/nov95/*.

the tclsh spot

by Clif Flynt

Clif Flynt is president of Noumena Corp., which offers training and consulting services for Tcl/Tk and Internet applications. He is the author of *Tcl/Tk for Real Programmers* and the *TclTutor* instruction package. He has been programming computers since 1970 and a Tcl advocate since 1994.

clif@cflynt.com

We all know the phrase "man is the tool using animal" (my wife claims "man is the tool buying animal"). We all tend to use the tools we know best for lots of purposes, whether they are actually the ideal tool or not.

I generally figure that tool overkill is not a problem. When I don't have a serial port analyzer handy, I've been known to debug RS-232 problems with an oscilloscope.

When I recently had a need to observe the HTML data flow between the browser and server, I started out with tcpdump. I've used tcpdump in the past to debug networking glitches, and I'm fairly familiar with it. Then I started trying to decipher the output.

Of course, the right tool for any job is Tcl, so that's what I used to extract the information I wanted from the tcpdump output.

The tcpdump program is fairly easy to install on SunOS and Solaris boxes and comes standard with most Linux distributions these days. On a SunOS system, you may need to play some driver games and run from root to use tcpdump, but it can be done.

There are a number of options you can use to customize tcpdump behavior. My favorites are:

-i interface	The name of the interface to watch.
-s size	The number of bytes of data to display as hex.
-l	Use line oriented buffering, rather than normal buffered I/O. Just because I'm not patient.
-n	Don't convert addresses to names.
-v	Be verbose. Include time-to-live and type-of-service information.
-x	Print each packet in hex.

The -x option makes a lot of applications possible. Being able to examine an entire data packet is a powerful tool.

The tcpdump output with this set of options resembles:

```
23:20:28.321319 < 192.168.9.63.www > 192.168.9.2.1823: P 1836:2062(226) \
    ack 329 win 8432 (DF)  (ttl 128, id 31790)
                           453c 2f74 683e 0d0a 3c74 6820 616c 6967
                           ...
                           2045 450d 0a3c 2f74 683e
```

The first two lines in this display are actually a single line in the tcpdump output, broken into two lines to fit better on the printed page.

The first line tells us that the packet was transferred at 11:20 p.m. from 192.168.9.63 (port 80) to 192.168.9.2 (port 1823). The packet had the "PSH" flag set in the TCP header, and includes 226 bytes of data.

The rest of the lines are the hex data for the TCP/IP packet including the IP header, TCP header and data.

The first easy step is to separate the packet information lines from the packet data lines.

The data format has a number of features we could use to distinguish information lines from data lines. For example, the first character for an info line is always a number, while the first character of a data line is always a whitespace character.

However, the first character of an info line can be any number, and I'm not sure which whitespace character (space or tab) is used for data, so testing on the first character would mean checking for one of 10 possible digits, or one of two possible whitespace characters.

The third character in an information line is always a ":". Checking for the ":" is a simple test. We can use the string first command to find the first colon in a line. If there is no colon, then string first will return a -1.

The Tcl code to read the data from an input channel and check for data or info lines looks like this:

```
while {[set len [gets $input line]] = 0} {
    # A colon in position 2 means a header line xx:yy:zz.abc
    if {[string first ":" $line] == 2} {
        # Found info line
    } else {
        # Found data line
        append hexData [string trim $line]
    }
}
```

The gets command will return the number of characters it reads from a channel. If there is a failure (like hitting the end of the file), it will return -1.

The string trim command will trim whitespace away from the left and right ends of a text string. This gets rid of the leading spaces. The whitespace was useful while parsing the input, but we don't need it anymore.

Whenever the script recognizes an information line, it knows that whatever is in the hexData variable is hex data for a complete packet, and this can be processed.

So, the next trick is to decipher that hex data and pull out the HTML page as easy-to-read ASCII text.

I like thinking of data as bytes, rather than 16-bit shorts, or 32-bit words. So, the first step is to convert the data from a string of shorts to a string of bytes.

Rather than think of the data as a string of numbers and spaces, it makes sense to think of it as a list of numbers separated by whitespace. This leads to thinking of list commands to reformat the data, instead of regular expression or string-based solutions.

Tcl has a couple of commands for converting data from strings to lists.

Syntax: join list ?separatorCharacter?

join	Converts a list into a character-delimited string.
list	The list of elements to join into a string.
?separatorCharacter	A character to place between each element.

The join command is very useful for converting a Tcl list into a character-delimited string to export to some other program. You can use the join command to create a comma-delimited string to export to an Excel or sc spreadsheet program, for instance.

If we declare the separator to be no character (a ""), Tcl will strip all the spaces from a list.

Rather than think of the data as a string of numbers and spaces, it makes sense to think of it as a list of numbers separated by whitespace

This code will convert the data string of space-delimited words to a long string of hex digits.

```
set hexData [join $hexData ""]
```

The next step is to convert the block of hex digits into a list of bytes.

The Tcl split command will split string data into a list.

Syntax: split string ?splitChars?

split	Split a string into a list. Elements are delimited by a marker character.
string	The string to split.
?splitChars?	A string of characters to use to mark elements. By default the markers are whitespace characters (tab, new line, space, carriage return).

The split command is often used to load data that was exported as a comma-delimited list from some other program like sc or Excel.

We can use the split command to split a string at each character by setting the splitChar to an empty string.

This code will take the mass of hex digits and convert it into a list of hex bytes.

```
foreach {h l} [split $hex ""] {
    lappend bytes $h$l
}
```

The packet consists of an IP header, a TCP header, and then the data. Our script should skip the header information and just process the data packet.

A TCP header is always 20 bytes long, but an IP header can vary in size.

The IP Header starts with two nibbles:

Identifier	The version number of the IP packet. Most commonly this is "4" for IP version 4. We'll be seeing "6" in this field more often as systems move to IPv6.
length	The number of 32-bit words in the IP header.

We can extract the IP header length with the string range and lindex commands.

Syntax: lindex list *position*

lindex	Return the list element at position.
list	A list of to extract an element from.
position	The position of the element to extract.

The first element of the hex bytes is the first byte of the IP header. It can be extracted as: [lindex $bytes 0], since Tcl lists and strings are zero based.

We could use the same split command to split the hex byte into nibbles, but it's a bit easier to extract one character with the string range command.

Syntax: string range string startPosition endPosition

string range	Return a subset of characters from a string.
string	The string to extract a subset of characters from.
startPosition	The position of the first character to extract.
endPosition	The position of the last character to extract.

The string range is zero based and uses inclusive selection, so the command to get the second character from the byte is:

```
set headerLen [string range [lindex $bytes 0] 1 1]
```

This header length is in 32-bit words, not bytes. The code to find the position of the first data value is:

```
set pos [expr ($headerLen * 4) + 20]
```

Now to convert the hex values to ASCII characters. As usual, there are several ways to solve the problem.

For instance, we could use an associative array as a lookup table with code like:

```
array set hex2ascii {
    ...
    41 A
    42 B
    43 C
    ...
}
...
foreach byte $bytes {
    append string $hex2ascii($byte)
}
```

This would work, but gets a bit large.

The Tcl format command works much like the C sprintf command, and makes a shorter and simpler solution.

Syntax: format formatString value1 ?value2?...

format	Return a new string formatted as defined by the formatString.
formatString	A string that defines how to format the following data units. This string uses the same conventions as printf. %i The value is an integer to be formatted as a decimal integer. %f The value is a floating point number to be formatted as xx.yy. % The value is character data to be formatted as a string. %c The value is an integer to be formatted as a character.
valueX	Values to be used in the return string. There must be one value for each "%" field in the format string.

The format command is most often used to generate tabular data with something like:

```
foreach {name address phone} $addressBook {
    puts [format "%20s %30s %10s", $name $address $phone]
}
```

We can use the %c option to convert the hex bytes to ASCII characters.

```
foreach b [lrange $bytes $pos end] {
    append str [format %c 0x$b]
}
```

Note the 0x$b. In Tcl (like C) a decimal number starts with a digit between 1 and 9. If a number starts with "0", it is considered to be an octal value, and if it starts with "0x", the number is treated like a hexadecimal value.

In Tcl (like C) a decimal number starts with a digit between 1 and 9.

Code like this:

```
set hex 0x41
puts $hex
```

would print 0x41, since puts is expecting a text string, and there's no need to consider 0x41 to be anything but a text string.

However, code like:

```
set hex 0x41
puts [expr $hex + 2]
```

will print 67. The expr command expects a number, and will interpret 0x41 as an integer (decimal 65), add two, and return the result as a decimal value.

The format command expects a numeric argument for the %c format specifier. When Tcl interpreter encounters the 0x$b it substitutes the current value for $b, creating an ASCII string like "0x41". The format command will interpret "0x41" as a hexadecimal value, and then convert the binary value to a printable ASCII character.

This code will convert the tcpdump hex dumps to printable ASCII.

```
proc hex2Text {hex} {
    set hex [join $hex ""]

    foreach {h l} [split $hex ""] {
        lappend bytes $h$l
    }

    set headerLen [string range [lindex $bytes 0] 1 1]

    # The IP header length is the second nibble.
    # The TCP header is 20 bytes.

    set pos [expr ($headerLen * 4) + 20]

    set str ""
    foreach b [lrange $bytes $pos end] {
        append str [format %c 0x$b]
    }
    return "$str"
}
```

The default tcpdump output includes every packet seen passing by an interface. This includes a lot of packets that I'm not interested in.

The tcpdump program has support for filtering the output by fields like port, IP address, type of packet, etc., which is great if you know just which packets you want to examine.

Sometimes I find that I want to grab all the data for a few minutes, save it in a file, and then examine different subsets of the data. This means that I need to filter out the unwanted data in my script.

The values I usually filter on are the source and/or destination IP address, source and/or destination ports, TCP flag and length of packet.

Most of this information can be extracted from the `tcpdump` info line with the Tcl regexp command, which was discussed in great and tedious detail in a Tclsh Spot arti-

cle two years ago. That article is online at: *http://www.usenix.org/publications/login/1999-12/features/tclsh.html*.

The basic form of the `regexp` command is:

```
Syntax:regexp ?options? expression string ?matchVar? ?subMatchVar?
```

Since a regular expression can include symbols that have meaning to the Tcl interpreter, we commonly put the expression within curly braces to disable any Tcl special character processing.

A simple regexp command might resemble:

```
set string {Regular expressions are useful and powerful}
regexp {R.*r +(e.*s) } $string all e1
```

The regular expression says to look for a pattern that starts with uppercase R, has 0 or more undefined characters, a lowercase r followed by one or more spaces, followed by lowercase e, 0 or more other characters, and finally a lowercase s followed by a space.

The parenthesis around the e.*s tells the regexp command to extract the part of the string that matches this part of the pattern and save that in the second variable.

When this command is run, it will match the expression pattern to the string Regular expressions. The entire matching string will be placed in the variable `all`, and the string "expressions" will be placed in the variable `e1`.

We can look for an IP address with a pattern like: {([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+)}. This pattern calls for one or more digits followed by a period, followed by one or more digits, etc. The periods need to be escaped with a backslash because the period has meaning to the regular expression parser – the period means any character. If we left out the backslash, then any collection of numbers would match the pattern.

We could match two IP addresses by putting two copies of that string (with appropriate other values to match the rest of the string) in a regular expression pattern.

In this case, the pattern starts to get very long, very quickly.

Another trick is to put the pieces of the regular expression into a set of variables, and then build the expression from those:

This code will look for the IP address, port, and flag information in a tcpdump info line and parse the values into five variables. The string of data from tcpdump is in the variable oldLine.

```
set addr {([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+)}
set port {.([^ ]+)}
set sep {[ <>]+}
set flag {([^ ])}

set m [regexp "$addr$port$sep$addr$port +$flag" \
    $oldLine all srcIP srcPORT destIP destPORT TcpFlag]
```

The string "$addr$portsepaddr$port +$flag" expands into {([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+).([^]+)[<>]+([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+).([^]+) +([^])}. Using the variables makes this a little easier to comprehend.

Once we've extracted that information, we can decide whether or not the data associated with this line is interesting.

This is an obvious place for an if command. The Tcl if command supports either a simple boolean expressions like $i < 10 or complex expressions like ($i < 10) && ($i > 4).

Since the boolean expression goes through the normal Tcl substitution phase, any Tcl command that returns a value can be used as part of the boolean expression.

We could check that a packet starts with the number 4 (the first nibble defines this to be IPv4) with code like:

```
if {[string range $bytes 0 0] == 4} {...}
```

We could check that one string matches another with one of several string commands like string compare (which returns whether one string is greater or less than another, like the C library strcmp function) or string first (that returns the first occurrence of one string within another). The string match command gives us the most power for this application.

Syntax: string match pattern string

string match	Returns a TRUE or FALSE depending on whether the pattern matches the string it is being compared to.
pattern	A glob style pattern to compare to a string.
string	The string to compare the pattern with.

A go/no go decision on each set of hex data is made with this code:

```
if  {([string length $hex]  $packetLen) &&
    ([string match 4 [string range [string trimleft $hex] 0 0]]) &&
    ([string match $destIPpattern $destIP]) &&
    ([string match $destPORTpattern $destPORT ]) &&
    ([string match $srcIPpattern $srcIP]) &&
    ([string match $srcPORTpattern $srcPORT ]) &&
    ([string match $flagPattern $TcpFlag ])} {
     set txt [hex2Text $hex]
     puts "\n$txt\n————————————————————-\n";
     }
```

This if statement checks that

1. The data packet is at least $packetLen bytes long.
2. The data is an IPv4 packet (or looks a lot like one).
3. The destination IP address matches the requested destination pattern.
4. The destination port matches the requested destination pattern.
5. The source IP address matches the requested source pattern.
6. The source port matches the requested source pattern.
7. The TCP flag matches the requested flag pattern.

The patterns could be hardcoded, but our code will be more configurable if the patterns are saved in a set of variables.

The body of this application looks like this:

```
# Define the patterns
set packetLen          100
set flagPattern        "P"
set destIPpattern      "192.168.9.63"
set srcIPpattern       "192.168.9.2"
set srcPORTpattern     "*"
set destPORTpattern    "www:"
```

```
set input                   stdin

# Initialize variables.

set hex ""
set oldLine ""
set TcpFlag ""
set destPORT ""

# Define the parts of the regular expression

set addr {([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+)}
set port {.([^ ]+)}
set sep {[ <>]+}
set flag {([^ ])}

# Read lines of data and process as necessary.

while {[set len [gets $input line]] = 0} {
    if {$len < 2} continue

    # A colon in position 2 means a header line xx:yy:zz.abc
    if {[string first ":" $line] == 2} {

        set m [regexp "$addr$port$sep$addr$port +$flag" \
            $oldLine all srcIP srcPORT destIP destPORT TcpFlag]

        if  {([string length $hex]  $packetLen) &&
            ([string match 4 [string range [string trimleft $hex] 0 0]]) &&
            ([string match $destIPpattern $destIP]) &&
            ([string match $destPORTpattern $destPORT ]) &&
            ([string match $srcIPpattern $srcIP]) &&
            ([string match $srcPORTpattern $srcPORT ]) &&
            ([string match $flagPattern $TcpFlag ])} {
             set txt [hex2Text $hex]
             puts "\n$txt\n-----------------------\n";
             }

        set oldLine $line
        set hex "";
    }   else {
            append hex "[string trim $line]"
    }
}
```

This code takes output from tcpdump and prints output that resembles this:

```
POST /cgi-bin/login.cgi HTTP/1.0
Accept: */*
Host: 192.168.9.63
User-Agent: Tcl http client package 2.3
Content-Type: application/x-www-form-urlencoded
Content-Length: 49

submitButton=Login&password=PASSWORD&loginName=loginID
```

As usual, the code described in this article is available at *http://www.noucorp.com*.

certificate revocation

Why You Should Do It and Why You Don't

by Paco Hope

Paco Hope has a M.CS. from the University of Virginia where he worked as the head system administrator in the Department of Computer Science. Hope joined Tovaris, Inc. in 2000 and is the director of product development.

paco@tovaris.com

Many companies implement X.509-compliant public key cryptography systems to identify parties in a variety of secure or authenticated communications. Public key certificates are used to secure email, grant access to the corporate intranet, verify secure Web sites, and perform a variety of other authentication and encryption duties. When public key certificates are presented, many integrity checks are performed on them to ensure their validity. Yet it is possible for all these checks to succeed even when the certificate is actually unfit for use.

Despite the fact that a certificate's signature is correct and the certificate has not yet expired, it may have been revoked. Certificates may be revoked for a variety of reasons and X.509 provides mechanisms for revoking certificates and learning about their revocation. Both Microsoft and Sun Microsystems have had to revoke certificates in recent years due to security breaches. Those revocations are supposed to provide information that prevents the end user from trusting the bad certificates. In practice, however, revocation mechanisms are rarely or poorly implemented. This article will introduce the concept of certificate revocation: what it means, how it happens, and how it fits into an overall public key infrastructure. We will explore why an organization wants to implement revocation, what solutions are out there, and some of the limitations to current practices.

Revocation Explained

X.509 certificates consist of various pieces of identifying information such as a name, email address, and serial number. They also contain the cryptographic key material used in the mathematics of cryptography. Their purpose typically is to bind a particular real-world entity (a person, machine, or company) to these mathematical bits for the purposes of some kind of secure transaction.

Early on in a secure transaction, an entity presents their certificate. To verify the authenticity of the certificate the other party performs several mathematical consistency checks on it. These involve checking the expiration date, verifying the digital signature that was applied by the issuing CA, and semantically interpreting various bits that indicate approved uses of the certificate. Those consistency checks, however, are not the whole picture. Certificates can pass all those checks and still be inappropriate to use. That's where the process of revocation fits in.

Revocation in X.509 infrastructures is solely the purview of the Certificate Authority (CA). Unlike PGP, where the end user usually revokes her own certificate, in X.509 only the CA who issued (signed) a certificate can revoke it. Revocation does not alter the public key certificate. After all, copies of the certificate might be stored in a browser cache, an email program's address book, or in various other convenient places. Instead, the fact that a certificate has been revoked is published in some other way. Applications which wish to check revocation status must first obtain the certificate, and then separately attempt to determine if the certificate has been revoked.

The first, and most well-known mechanism for publishing revocation is the Certificate Revocation List (CRL).[1] In their simplest form they are lists issued by CAs containing

the serial numbers of all certificates that the CA has revoked. A flexible and real-time protocol has also been defined and is offered by some major vendors. Other revocation methods, several of them patented, have also been defined and made available commercially. After first understanding the simplest CRL, we can explain the variations on that theme and explore other novel approaches.

Certificate Revocation Lists

Every certificate is issued by a CA, and that CA assigns it a unique serial number which is encoded in the certificate. The issuing CA and the certificate's serial number are paired to form an identifier for the certificate.

A Certificate Revocation List, then, is very much what its name implies. It is issued by a specific CA and lists the serial numbers of all certificates that CA has ever revoked. For each certificate it also lists the date the revocation occurred, and optionally a reason why the certificate was revoked. The entire list is signed by the CA's private key and presumably published widely. CRLs also include a publication date and a "next publication" date to indicate when the current CRL was issued and when the next CRL should be expected. By consulting the appropriate CRL, an X.509-compliant application can definitively determine whether or not a given certificate has been revoked.

In their original conception, CRLs would be published regularly by the CA, and made available through some public interface like FTP or HTTP. Finding CRLs this way would have to be supported by every X.509-compliant application – either directly or via the underlying operating system – from email software to Web browsers to VPN access software. PKI software, such as that which provides CA services, must also regularly update the CRLs in order to keep them current. Each application would initiate a connection each time it considers a certificate for which it has no verification information. Since CRLs regularly expire and are regularly updated, the application or operating system would have to monitor its revocation information and, if it became stale, refresh it.

The load imposed by HTTP or FTP connections from millions of desktop systems is one major disadvantage to this approach. If, for instance, secure email became a standard practice, and every recipient of every message were verified for revocation information, the burden of these connections would be a significant problem. Since CA certificates can live for very long times (some have expiration dates 20 years from now), CRLs can conceivably grow very long. Thus it could be especially inefficient to download and refresh a list of all revoked serial numbers, when the existence or nonexistence of a single number is the only information needed.

Improvements to CRLs

Several incremental improvements to the CRL process have been proposed and published; some have become commercial products. These improvements tend to focus on making CRLs easier to find, smaller to download, and/or easier to manage.

Entrust created and patented CRL Distribution Points (CDPs). A CDP is a reference (typically in the form of a URL) indicating the location of the CRL. This reference is included in the content of the certificate itself. Inclusion in the certificate makes it immediately available to any application which needs to find the relevant CRL. Entrust's Web site says they offer a royalty-free license to use CDP technology. Despite that, CDPs are rarely found in public key certificates. Even licensees like Microsoft and Verisign rarely – if ever – include CDPs in the certificates they issue to end users.

Delta CRLs are intended to make smaller, supplementary CRLs available more frequently, while making bigger complete CRLs available less often. In such a scenario the CA issues a base CRL at relatively long intervals, such as monthly or weekly. The CA additionally issues incremental CRLs that are supplements to the base CRLs. Those supplements (called "deltas") are issued on relatively short intervals, like daily or hourly. To definitively determine revocation, an application must have a sufficiently recent base CRL and the corresponding most recent delta CRLs. Delta CRLs offer modest improvements in performance, given their smaller downloads. The burden of many TCP connections to a central distribution location and the burden of making all applications or operating systems maintain CRL information remain substantially the same.

Still another variation allows partitioning of CRLs by indicating which CRLs contain which serial numbers. It allows the CA to manage the size of CRLs and allows clients to download smaller pieces of the overall CRL information.

Complete Departures from CRLs

A few other approaches to disseminating revocation information have been proposed. Certificate Revocation Trees (CRTs) form the basis of a Valicert product. They consist of complex binary hash trees computed by the CA. One primary advantage is that the revocation or lack of revocation can be represented by data substantially smaller than an entire CRL. Fractions of the overall revocation information can be sent to verify whether or not a certificate has been revoked. Like CDPs, CRTs are patented so there is only one product on the market that makes use of them, and it is not your Web browser or email software.

The IETF has standardized a real-time verification protocol for getting up-to-the-minute revocation information. The Online Certificate Status Protocol (OCSP) is defined in RFC 2560.[2] An OCSP "responder" is a system which listens on a network for revocation queries. Querying systems send a query identifying a certificate. That query may need to be digitally signed before the OCSP responder will honor it. The OCSP responder determines the status of the certificate in question and replies with that status, or indicates that the status is unknown. The OCSP responder must sign all responses using a special key and certificate issued by the CA.

The primary disadvantage to OCSP is the cryptographic demands. Every response must be signed, and the signatures on requests might have to be verified as well. This disadvantage can be overcome by sufficient hardware and network engineering. Like CRLs, however, if the certificate itself does not indicate the relevant OCSP responder, an application has no means of determining that an OCSP responder exists. OCSP responders have no way to "refer" a query, either. If a responder does not know the answer to a query, it has no means of indicating another server which might have the desired answer. Thus an application could issue queries to many different OCSP responders, but ultimately receive no valid revocation information.

OCSP is gaining momentum among large commercial PKI vendors as a more efficient method of determining revocation status. Widely available applications (Web browser, email software, operating systems) are slowly incorporating support for it. There are close to a dozen major vendors offering OCSP-based products, such as Digital Signature Trust Company, CertCo, and RSA Security. Making productive use of an OCSP responder, however, requires consistent and functional support in each and every X.509 application. Uniform and reliable OCSP performance will take time to mature.

Why You Care

Digital certificates and public key cryptography capabilities are rapidly appearing in all sorts of network-enabled applications. Every major email software program has some support for digital certificates. Most vendors of online chat and instant messaging services are adding cryptography to make secure chat possible. Internet e-commerce relies substantially on the integrity of certain CA certificates. Many VPN and firewall access systems utilize X.509 public key certificates in one way or another. What happens if a certificate is compromised?

We need only look to January 2001 to see a textbook example of why revocation is important.[3] Verisign erroneously issued two code-signing certificates to an unknown person who somehow persuaded them that he represented Microsoft. In a routine audit of January's activities, Microsoft noticed the erroneous issuance and the certificates were immediately revoked. This event was very well covered in security newsletters and publications. The fact is that someone somewhere has a certificate officially issued by Verisign that states unequivocally that it belongs to "Microsoft Corporation." Yet, the actual truth is that it neither belongs to nor represents Microsoft in any way. If you have not applied Microsoft's patch (which installs a partial CRL that covers the erroneous certificates), your Web browser or email software might automatically execute malicious code, because it has a seemingly legitimate, Verisign-issued certificate. The expiration date on those erroneous certificates is January 31, 2002. The malicious use of these certificates poses a threat long after that date.

Why You Don't Check Revocation

The simple answer is: you can't. CRLs are extremely difficult – if not actually impossible – to find. Verisign and Thawte publish their CRLs reasonably well. In this author's experience only Verisign's CRLs are so readily available that they are usable by applications that have only rudimentary X.509 support. There are many other CAs that I trust, however, and few if any publish CRLs that can be found by end users.

My Netscape browser (venerable at version 4.76) includes about 60 trusted root CA certificates. There is absolutely no association between any of them and their CRLs. The Web site I visit today may have a certificate that was originally issued and subsequently revoked by the "TC TrustCenter Class 2 CA." I will never know. Netscape (or Internet Explorer, or any number of other applications who might trust this CA) has no way of divining the location of the CRL for this CA. If the certificate I receive from the Web site happens to have a CRL Distribution Point in it, and the CDP points to a valid CRL, then I stand a chance. Otherwise, there is no way to know. If the certificate authorities will not publish their revocation information regularly and obviously, the end user has no hope of using it.

The usability of CRLs is so bad that email lists are needed to supplement them. When Sun Microsystems had to revoke two certificates in October 2000, they used email security bulletins to spread the word.[4] There was no CRL to rely upon. A telling detail is that the advisories identified the two revoked certificates, but they do not identify any mechanism of checking a CRL or any other regularly updated revocation information. Worse yet, Sun's security bulletin recommends deleting the certificate from browsers that have mistakenly trusted it. If the browser encounters the certificate again, it will prompt the user. The user will see "Sun Microsystems" and a valid signature, and will probably choose to trust the certificate. Deletion does not prevent the

The usability of CRLs is so bad that email lists are needed to supplement them.

REFERENCES

1. R. Housley, W. Ford, W. Polk, and D. Solo, RFC 2459, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile, January 1999. Status: PROPOSED STANDARD.

2. M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, RFC 2560, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP," June 1999.

3. Kathleen Murphy, "Verisign Gets Duped in Security Attack on Microsoft," *Internet World*, March 2001, *http://www.internetworld.com/news/archive/03262001b.html*.

4. Sun Microsystems, Security Bulletin 00198, October 2000, *http://sunsolve.Sun.COM/pub-cgi/retrieve.pl?doctype=coll&doc=secbull/198*.

browser from trusting the certificate. Only the presence of revocation information truly represents a barrier to trust.

Can you enable your existing enterprise applications or your operating system to make use of OCSP? The answer is probably yes. If you invest in a commercial OCSP offering of some kind, or if Microsoft or Apple or Sun integrate OCSP into their operating system you might get real-time revocation information. As it is, there are many concerns about OCSP's ability to scale, given its high demands for cryptography on every query and response. Various proposals are circulating to allow caching of answers, pre-signing of answers, and other mitigating techniques. Few, if any, of those improvements are readily available in both server and client implementations.

Conclusion

Revocation is a dirty little secret in the PKI world. Those who understand its role in secure transactions realize that current technologies simply fail to offer realistic solutions for the teeming millions. Some system administrators, CTOs, and decision makers are just getting their feet wet in PKI technology. They are probably already struggling to grasp the essentials — not to mention such subtle issues as how a certificate that appears valid is, in fact, invalid.

If you are considering investing in X.509 PKI technology, or if you have already invested the equivalent of a developing nation's economy in your PKI technology, ask some hard questions about revocation. Are your end users using some kind of technology (CDPs, OCSP, CRLs) to learn about revocation of your own certificates? Are your collaborators able to get up-to-date revocation information about your certificates? Are you able to get up-to-date revocation information about your collaborators' certificates? What if your collaborators bought from a different PKI vendor?

Consider all the different applications where you might be applying PKI technology. There are already many vendors hawking PKI-enabled email, Web browsing, B2B e-commerce, instant messaging, and VPN technology. How can revocation information be made available to each and every one of those applications?

Ultimately, security boils down to risk assessment and risk mitigation. Your organization will have to assess the risk of trusting a revoked certificate, and decide what amount of mitigation is required. The cost of that mitigation follows naturally from that analysis. Expect to make compromises. The ideal is to maintain and manage revocation information in all the places you use public key certificates. That ideal will probably not be realistic or affordable without some fundamental change in the way PKI companies and technologies operate.

high availability firewall/VPN with VRRP

by Dave Zwieback

Dave Zwieback is the technical director of inkcom (*http://www.inkcom.com*), a company that specializes in system, network, and security architecture.

zwieback@inkcom.com

Introduction

Internet connectivity has become mission-critical for many organizations, especially as they shift their connections to customers and partners from private lines to virtual private networks (VPNs). Not surprisingly, the availability requirements for Firewalls and VPNs have substantially increased. In this article, I discuss the implementation of a high availability (HA) Firewall/VPN using the Virtual Router Redundancy Protocol Monitored Circuit (VRRPmc). Specifically, I cover Check Point Firewall-1/VPN-1 version 4.1 SP3 running on appliances made by Nokia, with IPSO version 3.3. However, since most network equipment manufacturers currently support VRRP version 2 (RFC 2338) and VRRPmc, the general concepts discussed here apply to almost any HA network configuration.

Virtual Router Redundancy Protocol Monitored Circuit

If static routing is used and a router fails, users have to manually change their configuration to point to a replacement router. Using dynamic routing protocols (RIP, OSPF, BGP, etc.) allows for route replacement to happen automatically after a timeout. While dynamic routing clearly provides higher availability than static routing, dynamic configurations are more difficult to manage and can result in significant network overhead. In addition, transition from a failed router may take an unacceptably long time, a condition known as "black hole" periods.

VRRP is designed to combine the simplicity of static routing with the high-availability features of dynamic routing. In a VRRP Monitored Circuit configuration, users point to a static IP address of a virtual router. This virtual router has valid IP and MAC addresses, which under normal conditions are "owned" by the master router (i.e., the master forwards packets sent to the IP address of the virtual router and responds to appropriate ARP requests). In the event of master router failure, a standby backup router becomes master and assumes ownership of the virtual router (see Diagram 1). Typically, users experience no downtime during failure.

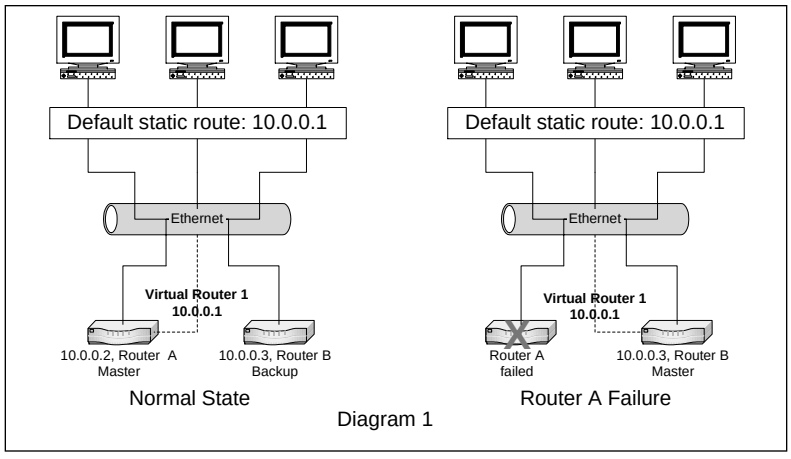
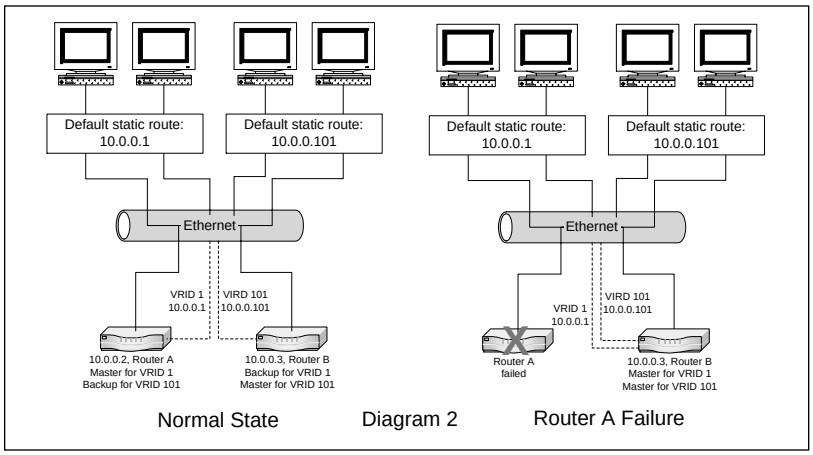


Diagram 1

Each virtual router must have a Virtual Router Identifier (VRID) in the range of 1 to 255. Although each virtual router may have more than one IP address, all the IP addresses of a particular virtual router are associated with one MAC address (from the address block assigned by IANA specifically for VRRP), and the VRID is the last octet: 00:00:5E:00:01:VRID. Thus, while two virtual routers with the same VRID can successfully exist on different LANs, VRIDs must be unique on a particular LAN to prevent MAC address conflicts.

A physical VRRP router may participate in (or "back up") more than one virtual router. For instance, a router may be master for one VRID and backup for another – a typical "active-active" configuration, illustrated in Diagram 2. This scenario is useful for load balancing between two (or more) routers, and offers increased performance in addition to high availability.



To determine which VRRP router is master or backup at any given time, each router must be assigned a priority value between 1 and 255 for each VRID. The router with the highest priority is the master. The master sends periodic advertisement messages to a special VRRP multicast address, 224.0.0.18. The frequency of these messages is typically 1 second, which can be changed by adjusting the "Hello Interval." When the master stops sending messages (for instance, in case of complete failure like a power outage), the backup router with the highest priority will take over the virtual router after a brief (<< 1 second) timeout.

For example, in an active-active configuration, detailed in Diagram 2, priorities can set be as follows:

	VRID 1	VRID 101
Router A	100	95
Router B	95	100

In this setup, Router A is master for VRID 1 and backup for VRID 101; Router B is master for VRID 101 and backup for VRID 1. Should Router A fail, VRID 1 is transferred to Router B, because it has the highest priority (95) of all routers participating in VRID 1 at that point. Since, Router B still has the highest priority (100) for VRID 101, the failure of Router A does not have any effect on VRID 101.

If a router with a higher priority than the current master comes online (for instance, a failed master router is fixed and becomes available), the virtual router moves to the router with the highest priority. Once again, users should not experience any downtime.

One of the most important differences between VRRPmc and VRRP (version 2) is how interface failure is treated. In VRRPv2, when an interface fails, only the failed one is transferred to the backup. On the other hand, in a VRRPmc configuration, the router can be configured to constantly monitor its physical interfaces (thus "monitored circuit"), and in case any of them fail, *all* the affected VRIDs are transferred to the backup. VRRPmc prevents asymmetric routing, which is required for the proper operation of the Check Point Firewall-1/VPN-1 in an HA configuration.

With either VRRPv2 or VRRPmc, the failover is accomplished by reducing the priority for the affected VRIDs by a specified amount (known as the "Priority Delta"). Once the priority of the master is lower than that of any of the backup routers, those with the highest priority immediately take over the VRIDs. Once again, the users should not experience a significant connectivity outage.

On a final note, VRRP is supposed to provide for strong authentication between participating routers, but currently the only choices on the Nokia platform are either no authentication or simple text passwords. It is highly recommended that the latter be used, especially in potentially "hostile" firewall environments.

Configuring VRRPmc on Nokia Appliances

The following is a sample Nokia appliance VRRPmc configuration, as detailed in Diagram 3. There are three networks: External, Internal, and DMZ. The two firewalls, A

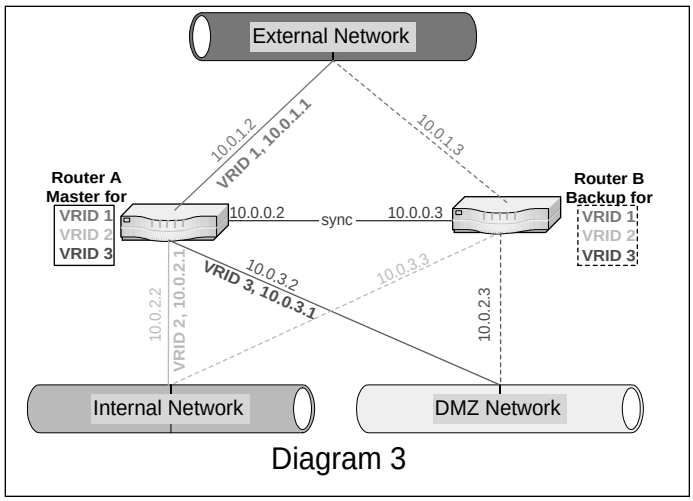


Diagram 3

and B, are connected to all three networks, and are also connected to each other via a crossover cable for Firewall-1 state synchronization (discussed below). Under normal conditions, Firewall A will serve as master for the three VRIDs (one on each network), while Firewall B will serve as backup.

The VRRPmc configuration can be accessed through the Voyager interface by going to Config→Router Services→VRRP menu. The VRRP settings for this configuration are summarized in the following table. Note that all the VRIDs are set up in monitored-circuit mode.

	Router A Master	Router B Backup
External Interface	10.0.1.2	10.0.1.3
Virtual Router (VRID)	1	
Priority	100	95
Hello Interval	1	
Backup IP (VRRP IP)	10.0.1.1	
Monitor Interfaces	Internal, Priority Delta: 10 DMZ, Priority Delta: 10	
Authentication	Simple Password: ViP1	
Internal Interface	10.0.2.2	10.0.2.3
Virtual Router (VRID)	2	
Priority	100	95
Hello Interval	1	
Backup IP (VRRP IP)	10.0.2.1	
Monitor Interfaces	External, Priority Delta: 10 DMZ, Priority Delta: 10	
Authentication	Simple Password: ViP2	
DMZ Interface	10.0.3.2	10.0.3.3
Virtual Router (VRID)	3	
Priority	100	95
Hello Interval	1	
Backup IP (VRRP IP)	10.0.3.1	
Monitor Interfaces	Internal, Priority Delta: 10 External, Priority Delta: 10	
Authentication	Simple Password: ViP3	

Each interface is configured to monitor two other interfaces (the only interface that is not monitored is the one used for firewall state synchronization). Thus, in case any of the monitored interfaces fails on Firewall A, the priority of all the VRIDs will be reduced to 90 (original priority of 100 minus Priority Delta of 10). Since the priority of the backup for these VRIDs is 95, all the VRIDs will be immediately transferred to Firewall B, which will become the new master. If Firewall A comes back online with priority higher than 95, it will take over the VRIDs once again.

Configuring HA Firewall-1/VPN-1 with Gateway Clusters

There are three steps involved in configuring the Check Point Firewall-1/VPN-1 for high availability:

1. Configuring the state synchronization
2. Configuring Gateway Clusters
3. Allowing VRRP traffic in the rule base

Step 1

Check Point Firewall-1/VPN-1 keeps track of all connections in its state table. In order to facilitate failover between two (or more) firewalls, this state information needs to be synchronized. On each firewall, the `$FWDIR/conf/sync.conf` file must contain a list (IP addresses or resolvable names) of all the firewalls that the state table should be synchronized with. In addition, a "control path" must be established between the firewalls, using the `fw putkey` command. Furthermore, since state information is exchanged approximately every 100 milliseconds, it is recommended that a separate network interface be dedicated to the task on each firewall and that time is synchronized between the firewalls as well, for instance, using xntp. A typical configuration is presented in Diagram 3.

With the firewall states synchronized, when the master fails, connections originally passing through the master continue through the backup uninterrupted. (`$FWDIR/lib/table.def` can be modified to include or exclude specific tables or protocols from state synchronization). One visible difference during the failover period is in the firewall log, where the *origin* of log entries will now be the backup instead of the original master firewall.

Step 2

With the introduction of Gateway Clusters, Check Point has considerably simplified HA firewall and VPN configuration. A Gateway Cluster is a virtual firewall, which consists of two or more physical firewalls configured for HA, for instance with VRRPmc and state synchronization. The steps involved in configuring Gateway Clusters are as follows:

1. Verify that the management console is separate from any of the HA firewalls (otherwise Gateway Clusters will not work).
2. Check "Enable Gateway Clusters" in the Policy→Properties→High Availability tab.
3. Create a Gateway Cluster object, with the external VRRP IP address: Manage→Network Objects→New→Gateway Cluster.
4. Modify the properties of each of the member firewalls to make them members of the Gateway Cluster created in step 3.

Once the firewalls become members of the Gateway Cluster, their individual Authentication, VPN, and Certificate settings disappear from their properties and can now be modified through the Gateway Cluster object. Furthermore, the security policy is now installed on the Cluster instead of on the individual member firewalls. However, by default, the policy install will fail unless it is successful on all the members of the Cluster. (This can be changed by checking off Policy→Properties→Security Policy→Install Security Policy only if it can be successfully installed on ALL selected targets, and by checking off Policy→Properties→High Availability→Install Security Policy on Gateway Cluster only if it can be successfully installed on ALL Gateway Cluster members.)

Traffic from the Gateway Cluster may be coming from either the "real" or the VRRP IP addresses. It is important to take this into account when creating firewall rules and objects. Specifically, in VPN configurations, if the remote firewall is using Gateway Clusters and is managed from a different management console, the remote firewall object should be created with the Cluster address and have all the "real" IP addresses for all the firewalls in the Cluster specified in the Interfaces Tab.

Step 3

As mentioned before, the master router periodically sends VRRP advertisements to 224.0.0.18. The firewalls should have a rule allowing VRRP advertisements:

1. Verify that the VRRP service is defined (with Match: ip_p=0x70).
2. Create the VRRP-MCAST-NET workstation object with an address of 224.0.0.18.
3. Create a rule in the security policy to allow VRRP traffic as follows:

Source	Destination	Service	Action	Track	Install On
MasterFW BackupFW	VRRP-MCAST-NET	VRRP	Accept	Long	FWCluster

As mentioned before, if you use the "Install On" field, ensure that Gateway Cluster object (FWCluster in the example above) is used instead of individual firewalls.

Monitoring and Troubleshooting

The VRRP status can be viewed from the Nokia Voyager, as well as from the command line, using iclid. Following are the iclid commands related to VRRP:

```
show vrrp                  quick summary report
show vrrp interface        interface configuration
show vrrp stat             vrrp stats
```

Furthermore, on the current master, ifconfig displays the VRID IP and MAC addresses along with the actual interface information.

```
FirewallA# ifconfig -a
…
inet 10.0.1.1/24 broadcast 10.0.1.255 vrrpmac 0:0:5e:0:1:1
inet 10.0.1.2/24 broadcast 10.0.1.255
…
```

By default, a VRRPmc IP address cannot be pinged, although this functionality can be enabled in IPSO 3.3. In addition to being useful for troubleshooting purposes, enabling this feature is required for certain routers and operating systems that will not forward any traffic to a gateway that does not respond to pings (for instance, the "dead gateway detection" in HPUX). Because it is not always possible to know all the specific

Using VRRP provides an easy way to greatly improve availability and performance of the network.

devices that utilize a particular router, it is recommended to enable "Accept Connections to VRRP IP" in the Voyager VRRP menu, and drop or reject *and* log icmp traffic with a firewall rule; this way, it is possible to identify devices that are pinging the router, which is extremely useful in troubleshooting.

Aside from configuration errors, most problems with VRRP installations occur when the backup router stops receiving VRRP messages from the master. This causes the backup to assume that the original master has failed, and to transfer to become master. However, if the original master did not actually fail and is still in master state, this "split-brain" situation may cause all sorts of havoc on the network. This condition is most often caused by VRRP traffic (to 224.0.0.18) being blocked, for instance by a firewall rule (or lack thereof) or an incorrect router or switch ACL or VLAN configuration. Please refer to Nokia Support Resolution 1521 for an excellent checklist of typical VRRP problems and solutions.

Furthermore, there are known issues with some Ethernet switches, mostly due to the fact that when a failover situation occurs, the MAC address (00:00:5E:00:01:VRID) of the virtual router "moves" from one switch port to another. The most common symptom is excessive VRRP transition master-to-backup time (sometimes over 30 seconds!), and thus a connectivity outage. Where appropriate, hubs can be used instead of switches, especially if the backup router is only used in standby mode. Otherwise, disabling MAC address caching and the spanning tree algorithm on the appropriate ports is required (on Cisco switches, `set port fast` will also work).

Conclusion

Using VRRP provides an easy way to greatly improve availability and performance of the network. Specifically, Nokia appliances running Check Point Firewall-1/VPN-1 can be easily configured for high availability with VRRPmc and Gateway Clusters. In addition to transparent failover in case of malfunction of one of the firewalls, the benefits of such a configuration include reduced maintenance and increased performance (in an active-active setup).

Bibliography

Check Point Software Technologies Ltd. 2000. Check Point VPN-1/FireWall-1 Administration Guide.

Knight et al., April 1998. Virtual Router Redundancy Protocol, Request for Comment 2338: *http://www.ietf.org/rfc/rfc2338.txt*.

Nokia Support: *https://support.nokia.com/*.

PhoneBoy's FireWall-1 FAQ: *http://www.phoneboy.com/*.

musings

by Rik Farrow

Rik Farrow provides UNIX and Internet security consulting and training. He is the author of *UNIX System Security* and *System Administrator's Guide to System V*.

rik@spirit.com

Funny how things run in cycles. Some economists and analysts had convinced themselves (and a lot of other people) that the business cycle of boom and bust had passed with the adoption of modern computer technology. And sniffing was supposed to be a thing of the past with the adoption of switches.

Password sniffing was endemic in 1996. Attackers loved installing password sniffers at ISPs. After all, an ISP sees all the traffic coming from many sites, and any login-name/password combination collected here will work through a firewall. The only thing preventing success would be source address access control through TCP wrappers or other software that has been properly configured.

ISPs (the smart ones) changed the architecture of their internal networks so that servers sat on their own subnets and didn't get to listen to all network traffic, just local traffic (which still included lots of POP passwords). Then people started installing switches, mainly to improve performance, and secondarily to help with security. Switches do what the names implies. Instead of broadcasting, like hubs, switches are supposed to provide a "switched" connection between ports, so that packet collisions are greatly reduced, and sniffing other systems' traffic becomes impossible.

Well, that was the hope. Truth is, switches may leak information. And switches can be attacked by flooding them with ARP packets, overflowing internal tables with new IP address/MAC address mappings until the switch goes back to acting like a broadcast hub. So switches did not turn out to be the panacea they were advertised to be. David Brumley's article in the November *;login:* describes some of the information collected from a sniffer at Stanford University.

Wireless

Wireless represents the newest networking mania. If you have been to a USENIX conference in the last several years, you will have noticed many people with laptops using the wireless network to communicate. USENIX provides a number of base stations (access points), all connected to a notebook acting as a router through to the Internet. This is very convenient, and I certainly appreciate it. Wireless has also appeared in businesses and homes everywhere. Not having to wire up your house or office has great appeal, and the newer versions of 802.11b support higher speeds and 128-bit encryption.

Too bad the encryption is pretty worthless.

Peter Shipley grabbed a lot of attention during this year's RSA Conference in San Francisco with a demonstration of war driving. Peter attached a microwave antenna to a wireless card in his laptop, added a GPS receiver, and drove reporters around downtown San Francisco identifying wireless networks.

You can do this yourself. Mark Langston did, but without the GPS and using only a Libretto sitting on his dashboard. You can read his account of war driving around the Silicon Valley area at *http://www.bitshift.org/wardriving.shtml*. Mark suggests that an even nicer way of doing this (well, more sinister actually) would be to use a Compaq iPaq running Linux, with a wireless card and external power supply, and simply leave it at a site where you would like to monitor the network. Even if the site is using encryption, the way the IEEE standards committee implemented the encryption leaves it open to many attacks.

WEP also includes a 32-bit CRC at the end of each encrypted data segment to provide a check on data integrity. This doesn't work either.

Really? Yep. There have been a raft of papers about cracking WEP (Wired Equivalent Privacy), the encryption that was supposed to make wireless equivalent to a wired network. The committee produced an excellent example of what happens when smart people, using proven crypto, design a system that fails – because they did not aggressively get the crypto community to check out their design.

Broadcasts

First off, war driving works because access points (and even laptop cards) broadcast management frames. Management frames are never encrypted, and include the Service Station ID (SSID). A lot of sites put their organization name there, making it really easy to identify an interesting network. Some access point hardware includes extensions for access control, but these are generally trivial to bypass. Lucent, for example, used the SSID as the password, making things really simple for Macintosh users, where the wireless software actually presents a list of SSIDs and asks which one to join.

802.11 included only 64-bit encryption using RC4, which sounds like it should work okay, even if the key space is a bit small. Many vendors have implemented what is known as 802.1x, with 128-bit passwords, and schemes for dynamically updating keys. And none of this works very well because of the design flaws in the standard, which everyone implements for interoperability.

First, a quick reminder about RC4. RC4 is a streaming cipher invented by Ron Rivest, licensed by RSA, and publicly available for about seven years. RC4 uses the key to generate a pseudo-random keystream, then XORs the data to encrypt with the keystream. To decrypt, all you need to do is to XOR with the same keystream. RC4 is used in SSL/TLS because it is fast and secure when used with a large key space (128 bits, for example).

So, what is wrong with WEP? 802.11 specifies a shared key for encryption. When using RC4, it is important never to use the same key twice since it is possible to perform cryptanalysis on two ciphertexts and to decrypt both without cracking the key. To avoid reuse of the single, shared key, WEP appends a value, the Initialization Vector, or IV, to 40 (or 104 for the 128-bit version) bits of the key. The IV is three bytes long, 24 bits, so there are over 16 million IVs possible.

WEP also includes a 32-bit CRC at the end of each encrypted data segment to provide a check on data integrity. This doesn't work either.

Ian Goldberg, of Berkeley, was one of the authors of a paper (*http://www.isaac.cs.berkeley.edu/isaac/*wep-*faq.html*) that discusses some of the shortcomings of WEP. Ian also spoke at the Blackhat conference in 2001, outlining several different ways of defeating WEP.

WEP networks can be configured to require authentication before a station can join. When a station contacts an access point, the access point sends a 128-byte challenge, and the station responds by choosing an IV, encrypting the challenge using the key created from the shared secret and the IV, and sending back the response. Now, anyone who has sniffed this exchange can XOR the challenge and response and use the resulting keystream (and the same IV) to authenticate to the same network.

The attacker at this point does not know the shared secret, but has authenticated. Since most people rarely change the shared secret, if an attacker can come up with the

shared secret, they can sniff/use this network for a long time. Most vendors support a key generator that takes a string and converts it to a key. Timothy Newsham (of @stake) discovered that the key generator itself is very flawed, throwing away most entropy, so that the resulting keyspace is only 2 to the 21 (about two million keys). An attacker can sniff a couple of packets, then use Newsham's tool to guess the key. During Blackhat demonstrations, Newsham correctly guessed several keys in just fractions of a second.

Well, that's not very good. You can input hexadecimal values for the keys instead (highly recommended), thus placing a brute force attack at a significant disadvantage. While the 40-bit secret can be cracked by 10 systems in a day, forget about brute forcing the 104-bit shared secret.

And there are other attacks. The simplest one described by Goldberg is to send ICMP Echo Requests, padded with the data of the attacker's choice, to workstations on the wireless network and sniff the packets. Now, the attacker has both the plaintext and the ciphertext and can recreate the keystream by XORing the two together. Theoretically, the attacker would have to do this for an entire 24-bit IV space. Practically, rebooting a system, inserting a wireless card, or entering a wireless network initializes the IV to zero, so collecting a keystream for all 16 million IVs won't be necessary.

Goldberg described three other attacks. In the double encryption attack, the attacker sniffs the network, waiting until the IV for a previously sniffed packet is about to be used by the access point. The attacker then sends many copies of the encrypted packet to a workstation on the network, sniffing again. When the IV matches, the keystreams will be the same, and the attacker can sniff the plaintext – the access point having decrypted it and transmitted it.

Because WEP uses RC4 and a communications checksum, it is trivial to modify a message. The attacker can XOR bits into the message, then XOR these same bits into the 32-bit checksum, and successfully modify an encrypted message. An attacker can take advantage of this attack to redirect modified copies of encrypted messages to a system the attacker controls just by changing the destination IP address (and perhaps the port address as well). When the packet passes through the access point, it gets decrypted and sent to the attacker.

Goldberg called the third attack a reaction attack. If the attacker suspects that a target has entered a password, the attacker can send spoofed packets with small modifications to the TCP header checksum to guess bits. When a guess is correct, the attacker will see an ACK or RESET packet. An incorrect guess results in no response. Guessing a password would require at least 56 guesses (one for each bit).

Cracking WEP

As if this was not bad enough, a group of mathematicians (*http://www.eyetap.org/~rguerra/toronto200/rc4_ksaproc.pdf*) postulated an attack that reveals the shared secret. They proved that RC4 reveals information about some bits in the key in the second encrypted byte (and others of the first 256 bytes as well). By capturing four or six million packets (depending on whether 64- or 128-bit encryptions was used), the entire shared secret could be deduced. Two SourceForge projects have software to deduce the keys. And while collecting millions of packets might sound ridiculous, 24GB drives are cheap, and that quantity of packets could be gathered from a busy network in a single day.

Because WEP uses RC4 and a communications checksum, it is trivial to modify a message.

The security implications of 802.11 have thoroughly convinced me that I will treat wireless like any broadcast medium, and only use it with SSH or some other form of VPN.

The IEEE 802.11 committee was made aware of many of the failings of WEP by Jesse Walker of Intel in October 2000. You can find his paper, as well as some of the others I mentioned, through a nice page of links, *http://www.cs.umd.edu/~waa/wireless.html*, which also includes a great paper by University of Maryland researchers that explains 802.11 in great detail. The IEEE will most likely do most of what Walker suggested, which includes increasing the IV length, using a block cipher instead of RC4, and using a cryptographic checksum that includes the keying material to prevent undetectable modifications to the ciphertext.

I had already decided to wire my house with CAT 5 before I learned all this about 802.11. Bill Cheswick had mentioned to me that he wasn't interested in turning his house into a microwave oven. While Bill was exaggerating, the security implications of 802.11 have thoroughly convinced me that I will treat wireless like any broadcast medium, and only use it with SSH or some other form of VPN. Mark Langston suggests putting access points in your DMZ – since the wireless network is effectively as much outside your building as it is inside.

I don't want to sign off without mentioning something that appeared at Netcraft, the Web surveyor's site (*http:// www.netcraft.com/survey*), on Slashdot, and at the Gartner Group's site. You have probably heard of the Gartner Group: analysts who get paid for their views about technology. On September 19, they posted an interesting opinion, which I am including just in case someone convinces them to retract what was posted at: *http://www3.gartner.com/DisplayDocument?doc_cd=101034*.

"Gartner recommends that enterprises hit by both Code Red and Nimda immediately investigate alternatives to IIS, including moving Web applications to Web server software from other vendors, such as iPlanet and Apache. Although these Web servers have required some security patches, they have much better security records than IIS and are not under active attack by the vast number of virus and worm writers. Gartner remains concerned that viruses and worms will continue to attack IIS until Microsoft has released a completely rewritten, thoroughly and publicly tested, new release of IIS. Sufficient operational testing should follow to ensure that the initial wave of security vulnerabilities every software product experiences has been uncovered and fixed. This move should include any Microsoft .NET Web services, which requires the use of IIS. Gartner believes that this rewriting will not occur before year-end 2002 (0.8 probability)."

Well, Code Red actually used the Indexing Server, which runs separately from IIS, so perhaps they should rewrite all of Win2K and XP. I can wait.

Wireless insecurity URLs:

A great page of links, including Jess Walker's, the Berkeley paper, and an explanation of 802.11: *http://www.cs.umd.edu/~waa/wireless.html*

Early release of software for cracking the WEP encryption in 802.11b: *http://sourceforge.net/projects/wepcrack/* (also */airsnort*).

The Fluhrer, Mantin, and Shamir paper explaining how to crack WEP: *http://www.eyetap.org/~rguerra/toronto2001/rc4_ksaproc.pdf*

IEEE Standard 802.11 standards, paper (by order) or PDF: *http://standards.ieee.org/catalog/IEEE802.11.html*

An interview with Peter Shipley about war driving: *http://www.starkrealities.com/shipley.html*

do you know what's in your firewall?

by Tom Limoncelli

Tom Limoncelli is director of operations at Lumeta Corp. and co-author with Christine Hogan of *The Practice of System and Network Administration*.

tal@lumeta.com

[Tom has shared with us some of the thoughts from his new system administration book. – RK]

I was surprised when Bob (not his real name) told me that he didn't have access to the rule sets installed on the firewalls at his company. He didn't mean that he didn't have easy access, or convenient access, or that it wasn't his job, or that he wasn't technically skilled enough to read a rule set. I mean he didn't have access. Nobody at his company did.

The company that he worked for, a major company whose name you would recognize, outsourced its firewall management to their Managed Service Provider (an MSP is an ISP with a larger feature set). This "managed service" was part of a package of convenient "value added" services. The ISP/MSP, I should mention, is also a name you would recognize.

How could customers not have access to their own firewall rules? The rule set is the critical list of filter rules that protects, or possibly fails to protect, a corporation's infrastructure! A minor typo is the difference between protection and Code Red.

The answer is simple. The MSP considered those rules to be their own intellectual property. Much to the customer's chagrin, the contract that they had signed agreed with this assertion.

I think outsourcing has many benefits. It can save money, it can let you focus on your business instead of the business of recruiting and retaining technical talent, and so on. However, you should always remember that when you outsource a task, you become responsible for checking the quality of the vendor's work.

My mother used to run a sandwich shop. She didn't bake her own bread. A local baker supplied bread to her and the other local restaurants. She was not responsible for making sure that the dough was properly mixed, prepared, baked, stored, and so on. Her job was quality assurance. When the bread arrived she had to make sure that it was the right type, quantity, and quality. Quality was the most important factor. Her customers took it for granted that she had the right type and quantity but she would lose business if the quality fell below expectations. If someone didn't like the bread, "I didn't make it" was no excuse. It was still bad bread. Of course, if she became dissatisfied with the bread, she could switch to another bakery.

The same is true for outsourcing services. You don't have to recruit employees, train them, and so on. You no longer have to mix the dough. Your job is now to make sure that the MSP is providing the quality of service that you need.

And that brings us back to Bob, my customer who didn't have access to the firewall rules that were protecting his company. I work for a company that makes a firewall rule set analyzer (see Wool, "Architecting the Lumeta Firewall Analyzer"). Bob hired us. After signing a Non-Disclosure Agreement, Bob went off to get his firewall rule set from his MSP so that it could be analyzed. He was denied.

If the customer wanted to know the rule set, the MSP argued, they should have kept track of every single change request submitted, guessed at what changes the MSP would be making as a result, and assumed their guesses were right. Of course, you paid for the MSP by laying off the person who could have done such duties. No offense, Bob.

An MSP has one very selfish reason not to reveal the rule sets in place: without the rule set, it's difficult for you to change MSPs.

An MSP has one very selfish reason not to reveal the rule sets in place: without the rule set, it's difficult for you to change MSPs.

There's another reason an MSP wouldn't reveal the rule sets. Chances are, they're very ugly. While the rule set began clean and pretty, the change requests that you've made were likely in the form of "Permit port FOO to host BAR" and "Block port FOO to host BAR." The easiest way to make such changes is always to add them to the front of the rule set. This requires very little thinking and is largely not prone to errors. However, after 500 requests, your rule set is 500 lines longer than the initial configuration. On the other hand, a carefully manicured rule set wouldn't be so large since each rule would have been installed with the kindest of care, with painstaking choice of where to insert the rule set for optimum performance, combining like rules to eliminate clutter, entered carefully at the keyboard by white-gloved hands while angels sing and cherubs toss rose petals into the air to create a beautiful and fragrant scene.

For an MSP to be profitable, there will be no white gloves, angels, or cherubs. Add the rule. Move to next customer.

In defense of MSPs, a rule set is intellectual property. There may be special filtering techniques in the rule set that are proprietary: how anti-spoofing is done, rules that permit special monitoring and Quality of Service protocols through, and so on. As part of any rule set modification, better firewall engineers do a highly sophisticated analysis that is on a par with the complexity of writing software. The MSP may have invested in special software that manipulates rule sets automatically.

All security-related systems need auditing. In *The Practice of System and Network Administration*, we discuss the benefits of self-auditing, and the very different role of external auditing, that is, audits by external groups. Both are needed. However, we never considered the ramifications of using MSPs. In this case, you are auditing yourself but really auditing someone else's efforts too. Auditing outsourced security services is just as critical, if not more critical, than auditing your own systems. Intellectual property issues must not get in the way!

What's the solution? Some companies have a policy that no firewall (or packet filtering device) will have write access by non-employees. This includes contractors, consultants, vendors, and ISP/MSPs. Such companies go to extremes. If their ISP places a router on their site, and the ISP requires access to said router, the company adds a router (or firewall) between the ISP's router and their network so that they have exclusive control of the filtering. This extra router can be expensive.

Alternatively, you can insert language in your contract that classifies the rule sets and configurations to be your intellectual property, to be revealed to you in a reasonable time frame, with financial penalties for non-compliance. If your MSP will not agree to that, at least put in the contract that you have the right to audit the configuration on a regular basis.

Having the ability to audit your firewall and actually doing the audits are two separate things. Establish a policy that sets down a schedule for regular audits, whether they are in-house or external. I know there is at least one fine company that provides this service. Check your business pages under "L".

Maybe we're dealing with the wrong problem. Maybe the problem is that we believe the promise of MSPs that offer soup-to-nuts solutions. They sound great but maybe

we should only let them do parts of the project: the installation, the software upgrades, and most importantly, the monitoring. Leave the policy for us to directly manage; while they may generate SLA (Service Level Agreement) statistics, it is our responsibility to validate and monitor those statistics. I once claimed jokingly that outsourcing works best when you outsource the boring parts (monitoring) but keep the fun parts (design and implementation) to yourself. Maybe that wasn't a joke.

If our auditing service does nothing but help companies realize they have signed contacts that hide their own firewall rule sets, we will have made the world a better place.

Ultimately, security is nothing more than risk management. Security for security's sake doesn't make sense. Business objectives (set by your CEO) must be translated to security policy (set by your CIO or someone who reports to your CIO), which should then be translated into firewall rule sets, access systems configurations, host configurations, and so on. Trusting someone else to manage your firewall is a risk, and it may be an acceptable risk based on the business objectives of your company. Blindly trusting someone to do this without having the ability to audit their work is both dangerous and irresponsible.

The next time someone tries to sell you MSP services with no right to audit the configuration, sit them down and tell them about my mother's sandwich shop.

P.S. Bob did eventually get the rule set out of his MSP. It required a three-way Non-Disclosure Agreement to be signed between us, the client, and the MSP. The audit then proceeded without a hitch.

Bibliography

Avishai Wool, "Architecting the Lumeta Firewall Analyzer," *Proceedings of the 10th USENIX Security Symposium*, August 2001, Washington, D.C.

Thomas A. Limoncelli and Christine Hogan, *The Practice of System and Network Administration*, Addison-Wesley, 2002, ISBN 0201702711.

if computers had blood, we'd be called doctors

SAs versus MDs, Part I

by Steven M. Tylock

Steven Tylock has been managing infrastructures for the past 15 years in the Rochester NY area. He helped organize GVSAGE as a local SAGE for the region,and has promoted GVSAGE talks at the ITEC technology show.

stylock@gvsage.org

The basis for this article is the many and varied analogies that I have seen applying medical terminology to our world of computers. I first distilled these into a paper submitted to the LISA conference – the paper was not accepted (it's tough to get non-technical papers past the panel), but many of the reviewers liked the substance and encouraged me to reformulate it into a *;login:* article. Part one of this article draws parallels between the world of medicine and the world of computers. Part two explores the growth of the American Medical Association (AMA) as a society of medical professionals and compares it to SAGE as a society of computer professionals.

Consider the parallelism between the professions. Computers have a life cycle from construction (birth)[1] through end-of-useful-life (death) all the way to the reclamation of board-level materials (decomposition). In between, both professions deal with upgrades (growth), component failure (illness), and even component replacement (organ transplants). Some of the language is common to both professions: virology, for example. In medical and computer care, it is best to prevent illnesses and other problems, but emergency care is still very important. Traditionally, the bedside manner has been a defining characteristic of "good doctors"; system administrators with people skills as well as technical skills are similarly well regarded and sought after.

Specialization

The field of medicine is highly divided and specialized, but even with this specialization, the general practitioner is still a valued member of the field. We can consider the general internist or family practitioner on the side of the generalists. On the other side, we can consider the specialists: surgeons, pediatricians, ob/gyns, dentists, podiatrists, dermatologists (we can't even begin to list them all in the space provided here).

The levels of ability run from the technical know-how of the radiology technician to the years of training required to get a license as a Doctor of Medicine.

Beyond correcting medical problems and keeping people healthy are lots of other medical endeavors, including research into the nature of life, exploration of disease, discovery of how things work, and even attempts to augment nature with mechanical aids.

System administration is similarly configured. The generalist might go by the jack-of-all-trades moniker "system administrator" or any number of equally vague names. Some specialists note that they administer the network, database, phone system, or a specific type of server or application software. In the same way that you wouldn't ask your eye doctor about having a baby, you won't ask your Notes administrator about problems with your router.

The levels of ability run the gamut from support technician to guru. Currently the field does not license professionals, but that may change as time goes on.

And beyond the bounds of installing and maintaining systems are many other endeavors to research components, create operating systems, discover how things break, and improve the way we build, maintain, and use these systems.

Methods of Care

People's approach to their health care varies. Some people don't get any care, avoiding medical institutions entirely. Some try to take care of themselves (potentially with limited knowledge), and others get advice and products from modern-day snake-oil peddlers. Some find care from well-intentioned but unlicensed suppliers, and some get care from licensed medical professionals.

The medical field attempts to match people's needs with different levels of service. Hospitals provide acute care for those with life-threatening problems; they also provide ambulatory and ancillary care with their specialized environments. Clinics provide preventative and non-acute care for individuals servicing non-office hours, and primary care physicians aim at preventative and non-acute care.

The computer field has its own wrinkles but is remarkably similar. People ignore their PC problems, try to tinker with them themselves, take advice from people and places they have never heard of before, and call in professionals to help. It is similar to the days when doctors were not licensed by the state – any person claiming to have the knowledge can hang a shingle up in front of their barber shop and call themselves a computer support specialist.

Coincidently, system administrators would rather build an environment where problems are recognized early and corrected before they become big problems in the same manner that physicians work to prevent disease. Sysadmins track their activities with sophisticated trouble-ticking tools and employ triage in the field as well as scheduled trips to the back office for reconstructive work. Like doctors, sysadmins use pagers and cell phones to stay "on call" in case emergencies develop that require their efforts.

Training

I'm going to end this section with a discussion of training – again the comparisons are surprisingly consistent. The medical profession has tuned its training methods to include both theoretical and practical experience; SAGE should consider this as it moves forward.

Many sysadmins can trace their roots to a student/mentor or apprentice/master type relationship in their past. No courses existed to provide the required training; learning by watching and doing was the best and only method. In much the same manner, early doctors relied on the apprentice relationship:

"In the days when apprenticeship was the mode of medical education, for example, the apprentice learned what the doctor did and why, observed how he did it, and then went on to practice in that fashion."[2]

Two additional comments in the same book match experiences in the sysadmin realm:

"Everyone who becomes a specialist of any kind prepares in two ways: (1) he reads and attends lectures; and (2) he is influenced by observations of the people already in the field."

"If medicine as a profession and medical education require thought and judgment, unless one understands 'how' and 'why,' one will not necessarily react suitably in complex situations."

These describe the academic versus real-life approaches to training, a common thread in sysadmin training theory; someone may know all of the steps required to perform a

It is similar to the days when doctors were not licensed by the state – any person claiming to have the knowledge can hang a shingle up in front of their barber shop and call themselves a computer support specialist.

REFERENCES

1. Dell will give you a birth certificate on its Web site: *http://support.dell.com/us/en/birth_certificate.asp?st=test*.

2. George A. Silver, M.D., *A Spy in the House of Medicine*, pp. 131-132. Aspen Systems Corporation, 1976.

3. M. Kaufman, "American Medical Education," in R.L. Number, ed., *The Education of American Physicians*, p. 27. (Berkeley: University of California Press, 1980).

4. T. Darmohray, ed., *Job Descriptions for System Administrators,* p. 1. (Berkeley: USENIX Association, 1997).

5. W. LeFebvre, ed., *Educating and Training System Administrators: A Survey*, p. 17. (Berkeley: USENIX Association, 1998).

6. P. Starr, *The Social Transformation of American Medicine*, pp. 123-124. (New York: Basic Books, 1982).

7. H. Spencer, "How to Steal Code, or Inventing the Wheel Only Once," in *USENIX Association Conference Proceedings*, Winter 1988.

certain operation yet be clueless in actual situations. If sysadmin lore recalls an approach where the only learning offered was through apprenticeships, and current sysadmin diploma-mills highlight a current approach of learning through bookwork, future success in the field would seem to require both elements.

Martin Kaufman extends this concept further by considering the social factors involved:

"In sum, it can be said that medical education has changed greatly over the years, but the attempt to develop perfection has failed in every period. Indeed, although a change in the focus of medical education itself may have appeared at the time to have been a revolutionary improvement, in retrospect that great advance often brought with it a new set of problems. For instance, although the modern medical student is trained in more ideal physical surroundings and by better professors than his eighteenth-century or nineteenth-century counterpart, he has little of the personal contact that typified the apprenticeship."[3]

Anecdotally, sysadmins know that neither education nor experience alone is a predictor of success as a system administrator. Sysadmins are fond of exchanging stories – of "papered" sysadmins who couldn't find their way out of a paper bag to self-taught sysadmins who lacked the grounding a fundamental education brings. SAGE itself hits on this in the short-topic booklet on job descriptions and education and training:

"Most get their skills through on-the-job training by apprenticing themselves to a more experienced mentor. Although the system of informal education by apprenticeship has been extremely effective in producing skilled systems administrators, it is poorly understood by employers and hiring managers, who tend to focus on credentials to the exclusion of other factors when making personnel decisions."[4]

"Almost all of the instructors who were contacted during the writing of this booklet mentioned the importance of hands-on experience as a component of learning system administration. Several of the instructors also mentioned that it was difficult to provide bona fide hands-on experiences in a classroom or even a laboratory setting."[5]

The medical profession's use of intern and residency programs that began at the end of the nineteenth century[6] may not hold a complete answer to the dilemma faced in the sysadmin world, but it bears consideration for a formal education interspersed with apprenticeship-type work in a real environment.

Where To Next?

By examining the history of American medicine, we continue with a philosophy that sysadmins have kept over the relatively short life of our profession – stealing shamelessly from things that work,[7] and learning from things that don't. In the next segment of this article, we'll look at the growth of the AMA; don't feel bad – it took them decades to develop into the organization that they are today.

ISPadmin

Web Hosting

Introduction

In this installment of ISPadmin, I examine how ISPs implement their Web infrastructure to support retail (tilde, or "~", accounts) and hosted domains. Web hosting is an integral part of most if not all ISPs, and many companies (Rackspace Managed Hosting and ServInt to name two) focus exclusively on Web hosting as their core business. Web hosting was the first application to be offered in the area now known as "application hosting."

by Robert Haskins

Robert Haskins is currently employed by WorldNET Internet Services, an ISP based in Norwood, MA. After many years of saying he wouldn't work for a telephone company, he is now affiliated with one.

rhaskins@usenix.org

It is worthwhile discussing the typical migration of a Web hosting customer at a retail ISP. A traditional dial ISP customer starts out buying a "standard" dialup account, which usually consists of the following:

1 dialup (PPP) account
1 mail (POP) account
1 Web hosting account (of the form *www.isp.net/~username*, commonly referred to as the "tilde" account)

Figure 1 illustrates a typical migration path of a Web hosting customer. The subscriber starts utilizing their standard PPP account, and the "tilde" Web account if they want to have some sort of a Web presence. If it is a business account, or a retail subscriber with more than a passing interest in hosting Web content, they will probably outgrow the "tilde" account and want to move to a "real" hosted domain (*www.mydomain.org*). The ISP needs to have a Web hosting offering or else they will lose the customer and associated revenue.

Once the hosted domain owner needs to sell something, they will want to have a shopping basket, secure site, credit card payment mechanism, etc. Once again, unless the ISP wants to risk losing the business, they need to make sure they can support electronic commerce.

The final step is the case where the Web site owner has so much traffic, it needs to be hosted on a dedicated server. To keep this customer and their business, the ISP must have a collocation (colo) offering.

In many senses, the Web hosting business is just an offshoot of the real estate business. In order to support a large number of domains, data center space is required. Of course, there is more to Web hosting than just real estate (for example, UPS and backup generator power, fire suppression, network connectivity, monitoring, etc.), but a large component of the cost will be the "bricks and mortar" and similar fixed non-IT-related components.

Web Hosting Infrastructure

In most cases, a small provider would likely have a very similar setup to a larger provider, but it might differ by (1) using a shared machine, in which the Web hosting machine might also perform other functions (such as mail, RADIUS and/or DNS) and (2) having fewer automated Web hosting-related provisioning and billing mechanisms.

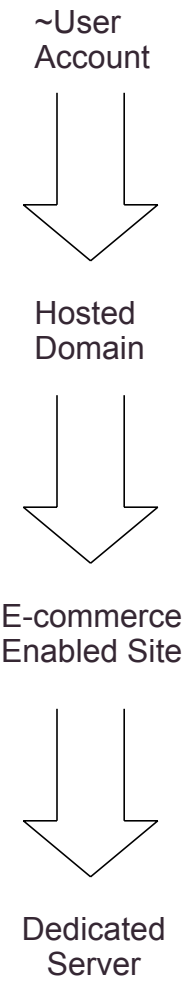


Figure 1

It is useful to have some idea of what it takes to manage a Web infrastructure for a dialup provider.

A large ISP would likely have machines dedicated to each specific type of hosting. For example, a machine or series of machines would be dedicated to each of the following functions: "tilde" accounts, domain hosting, and e-commerce-enabled domain hosting.

It is useful to have some idea of what it takes to manage a Web infrastructure for a dialup provider. Ziplink hosted approximately 5000 "tilde" accounts on a Sun Ultra 10 with 9GB mirrored disk drives. For the hosted domain side of the business, there were two Sun Ultra 10s with mirrored 18GB drives, each server hosting approximately 150 domains. The load, under normal circumstances (i.e., with no bad CGI scripts running and no extremely active pornography sites) was always less than 0.5. This infrastructure was run by the equivalent of half a full-time mid-level staff member. This might seem high, but Ziplink did not have much automation in the area of Web hosting, since the business plan focused on other areas (namely, wholesale dial up).

Web Server Software

Apache

By far the most commonly utilized Web server, Apache has outstanding support for service providers and is very configurable. Some of the Apache modules I have found particularly useful at ISPs include: mod_alias, mod_rewrite, mod_userdir, mod_speling (particularly useful for migrating a VMS or other non-case sensitive O/S Web server to UNIX), and mod_vhost_alias.

The references at the end of the article contain a link to the virtual domain support page for Apache.

Microsoft

Providing Microsoft (MS)-based services (such as Active Server Pages [ASP] or FrontPage Extensions) requires some extra effort if the ISP is a UNIX-centered shop. Some service providers may charge more for an NT-based infrastructure. UNIX-based service providers have two options. They can either (1) install a Microsoft-based infrastructure or (2) install additional software components to enable the required Microsoft functionality.

If option 1 is taken, a parallel MS Web hosting infrastructure must be deployed. This configuration would take the form of a Windows 2000 (Win2K) server running IIS (which now ships as part of Windows 2000). A stock Win2K server (aka NT) would be able to handle both ASP and FrontPage Server Extensions (FPSE) functionality. (Note that one must download and install the FPSE for the Microsoft platforms; Win2K and NT do not ship with FPSE). If the ISP has a back-end billing/provisioning system, then such system(s) must be modified to provision and bill this infrastructure.

If option 2 is taken, Apache (and most non-Microsoft-based Web servers) require additional components. These additional components can often be engineered on top of the provider's existing Web infrastructure. In order to support FPSE, Microsoft has a software package which in its current version (FrontPage Server Extensions 2002 for UNIX) runs under Apache 1.3.19 and implements the server side of FrontPage. In order to implement MS ASP functionality under Apache, a package such as Sun Chili!Soft ASP must be implemented. There is also a Perl ASP (with Perl scripting only) implementation for Apache available called Apache::ASP. I do not have any direct experience with either of these packages.

The other popular Web servers out there (iPlanet and Zeus) are not often deployed at service providers, probably because they are both commercial products, unlike the public-domain Apache.

The Netcraft site has a good graphic showing the statistics of the various Web servers out on the Internet. I was particularly interested in the July 2001 report, which showed a rather large downturn (4.29%) in the number of Apache sites and a significant upturn (5.49%) in the number of Microsoft IIS installations. According to the Netcraft analysis, this is due to two large sites converting to Microsoft, and has been masking a larger trend away from Apache to IIS. The August 2001 report showed a much smaller increase in IIS deployments, so the July 2001 increase appears to be an anomaly.

Web Server Issues

There are many challenges facing service providers who host Web content. I will briefly discuss the issues and solutions surrounding each.

Managing Domains

Managing a large number of domains is a problem. Most larger Web hosting companies have a file-naming scheme whereby the top-level namespace is broken down by the first letter of the domain. Thus, directory /www1/a would house all domains starting with "a", directory /www1/b would house all domains starting with "b", and so forth. Of course, the top-level directory name might indicate the machine name (www1, www2, etc.) for individual Web server machines). Also, Apache's URL mapping scheme can be utilized to automatically redirect domains to the correct Web server domain content via a "global" command, without the need for specific per-domain configuration entries in the Apache configuration file.

Email Aliasing

"Email aliasing" refers to the ability for email at a hosted domain (*info@mydomain.com*) to be forwarded to a "real" mailbox, for example *cust@isp.net*, where the customer actually picks up the mail through POP3, IMAP, Webmail, etc. Both the Sendmail and Postfix mail transport agents have good support for virtual mail configurations. The ISP usually adds an interface (typically Web based) for the customer support agent and/or the customers themselves to edit these mail mappings. These mappings must be forwarded to external accounts or the service provider's "regular" dialup mail accounts. As a result, there is usually an interface between the provider's billing system and mail infrastructure.

Legal Liability of Hosted Content

Most providers consider themselves "common carriers" and do not police the content placed on Web sites by their customers. Of course, people can and do complain. If it is an obvious copyright infringement (for example, posting illegal software or copyrighted mp3s) or something similar, the provider usually can and does take swift action without waiting for a court order. However, if the complaint is not as clear cut as that (for example, a site that parodies someone or something), the provider usually will wait for a properly executed court order before taking action. Most if not all service providers do not monitor content, since the provider would then be expected to monitor all content. Please be aware that policies in this area vary quite a bit from one provider to another so it is dangerous to make too many generalities!

Managing a large number of domains is a problem.

Most providers have automated mechanisms for domain registration as well as signup for Web hosting service.

Security

Security of the server as well as security of customer data can be a problem in a shared Web server environment. Many hosting providers will not allow arbitrary customer-written/provided CGI scriptsto be run on the machine. Of course, CGI routines are usually made available for standard functions like Web counters, comment sections, Weblogs, etc. However, any non-standard code has to be reviewed by staff prior to implementation. Also, external programs such as CGIWrap are used to help ensure CGI programs are run in a secure manner. The normal file access control mechanisms of the host (UNIX or NT/Win2K) are used as well.

Logs

Most ISPs generate access and error logs for their customers. Apache has excellent support for automatically generating these logs without human intervention. The Apache configuration commands used to generate per domain logs are "ErrorLog" and "CustomLog" and appear under each VirtualHost section on a per-domain basis. Access to the logs is usually granted via the same FTP interface the customer uses for uploading their content.

Bandwidth

Most Web hosting plans include limits on the amount of bandwidth each customer's pages generate. This bandwidth accounting is meant to limit the ISP's exposure if a customer's site should suddenly become very popular (for example, the customer begins to host pornography). If a customer's site does become too much of a load on a shared server, the service provider will request that the customer move to a dedicated server. Of course, the ISP will charge additional money for that additional functionality to cover costs. These costs are: server hardware (if provided by ISP), staff time (to set up and move the domain) and transit (network bandwidth to the Internet).

Billing Integration

A small ISP will usually do everything (setting up DNS, configuring Apache, etc.) by hand; as a result, no billing integration is required or possible. Billing for a larger ISP or Web hosting provider would be much more integrated, as Web hosting might be a larger part of the ISP's business. Most providers have automated mechanisms for domain registration as well as signup for Web hosting service. This would require an automated interface into the provider's billing system. How this interface is achieved would be a function of the billing system as well as the provider's business model.

E-commerce

For the purposes of this article, e-commerce (electronic commerce) includes: a shopping basket, managing an inventory, and processing credit cards. The ISP that wants to retain its customers through the full life cycle must have e-commerce capability. Because of the financial risks something like this poses, e-commerce could be farmed out to a third party or could be done in-house via commercial software or open source software.

Many third-party providers have a complete e-commerce package. In addition, several business to business (B2B) and business to consumer (B2C) solutions exist for ISPs. The most commonly implemented open source shopping basket and inventory management software is RedHat Interchange (formerly Akopia Interchange). Interchange is a full-featured B2C software application written in Perl and is widely used. Credit

card processing is dictated by the kind of e-commerce software the ISP is using. RedHat Interchange, for example, supports many credit card payment gateways, including Cybercash/Verisign (Verisign recently acquired Cybercash).

Next time I'll take a look at how ISPs design their backbone networks. In the meantime, if you have a question about ISPs, wondered why something was done a particular way, I'd love to hear from you!

I'd like to thank Vinny Bono of GlobalNAPS for his input.

References

Apache modules: *http://httpd.apache.org/docs/mod/index.html*

Apache Virtual Host documentation: *http://httpd.apache.org/docs/vhosts/index.html*

Apache Web server start page: *http://httpd.apache.org*

Apache::ASP: *http://www.apache-asp.org*

CGIWrap: *http://cgiwrap.unixtools.org*

Cybercash/Verisign: *http://www.verisign.com*

Digex: *http://www.digex.com*

iPlanet: *http://www.iplanet.com/products/iplanet_Web_enterprise/home_2_1_1m.html*

Microsoft FrontPage Server Extensions download: *http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnservext/html/fpovrw.asp*

Microsoft FrontPage: *http://www.microsoft.com/frontpage*

Microsoft IIS: *http://www.microsoft.com/iis*

Microsoft Web page: *http://www.microsoft.com*

Netcraft Web Server Survey: *http://www.netcraft.com/survey*

Perl: *http://www.perl.com*

Postfix: *http://www.postfix.org*

Rackspace Managed Hosting: *http://www.rackspace.com*

RedHat Interchange: *http://interchange.redhat.com*

Sendmail virtual domain documentation: *http://www.sendmail.org/virtual-hosting.html*

ServInt: *https://www.servintservers.com*

Sun Chili!Soft ASP: *http://www.chilisoft.com*

Zeus: *http://www.zeus.co.uk*

stepping on the digital scale

by Erin Kenneally

Erin Kenneally is a Forensic Analyst with the Pacific Institute for Computer Security (PICS), San Diego Supercomputer Center. She is a licensed Attorney who holds Juris Doctorate and Master of Forensic Sciences degrees.

erin@sdsc.edu

Duty and Liability for Negligent Internet Security

The Fine Line: Are You a Victim-Symptom or Liable-Cause?

Reality dictates that networked computers are vulnerable to undesired actors and events owing to computer security vulnerabilities. The important question becomes: should these insecure computers be tolerated given the nature of modern computing infrastructure? If the answer is "no," we must be prepared to define standards of care in securing network computers against damage by third parties and recognize the recovery of damages from those parties whose insecure computers were used to exact harm.

Much of the popular media has focused attention on the ever-growing incidence of insider malfeasance and external intrusions into computer systems, resulting in violations of privacy, network failures and disruptions, spread of viruses, fraudulent transactions, and corporate espionage and data tampering, among others. A comparable amount of publicity has been paid to identifying the cybervandals, kiddie hackers, angry customers, disgruntled employees, foreign moles, or unethical competitors. Indeed, the panoply of crimes and damaging activity carried out over computer networks (Internet included) has both emulated and broadened the miscreant feats of property-based society.

Similarly, it is not surprising that a litigious society confronted with these expanded criminal capabilities and opportunities for mischief will spawn a wide array of legal redress seekers. To be sure, the cyber-equivalents of the McDonald's coffee-scalding lawsuits and Twinkie defenses have emerged and will likely persist. When the digital perpetrator cannot be tracked or is insolvent, the wronged party will seek alternate entities to hold responsible for losses incurred. To this date, no court has squarely addressed the issue of liability for failure to adequately secure a computer system. But, insofar as computer security technology represents the means to thwart these harmful activities, the logical targets from which to seek redress are those parties that have failed to implement appropriate computer security practices.

Our legal system exists to provide a mechanism of protecting individuals' interests and resolving disputes in an effort to maintain an orderly society. It is guided by notions of reasonableness and judged by objective standards representing society's values. In this sense, laws – both legislative and judicially created – are formal embodiments of society's willingness to assign responsibility and redress grievances between parties. However, the nature of our computer-networked environment forces society to redefine what is reasonable and fosters responsibility shifting. The critical question entails assessing responsibilities, defining duties, and assigning liabilities amidst this novel playing field that cultivates both traditional and neoteric relationships, conduct, and consequences. It is against this backdrop that this article discusses potential liability for "computer insecurity" between software vendors (SWVs), service providers (ISPs/ASPs), Web businesses (WebCos), and individual users within our computer-networked society.

This article highlights the parties and common scenarios likely to spawn claims of negligence for failure to secure computer systems. Is there a duty to secure computers?

Where does that duty arise from? To whom does the duty apply? What is the scope of the duty? Should "insecure parties" be assigned different levels of duty regarding computer security? What does it mean to take reasonable precautions to prevent computer intrusions?

What does it mean to take reasonable precautions to prevent computer intrusions?

The Legal Playing Field – Enter Negligence Claims

Negligence is primarily a concept within civil law, which is intended to address grievances between people and encourage socially responsible behavior. This is in contrast to criminal law's purpose of enforcing the government's interest in deterring future crime by punishing perpetrators. When a user or business suffers loss from an invasion into their computer system or network, criminal law offers no compensation to the victim if the intruder cannot be identified and/or is judgment-proof. This is more of a rule rather than exception given the ability to act anonymously, difficulty tracing the origin of malfeasance, and perpetrator profile (i.e., juvenile miscreants) associated with the Internet.

Similarly, contract law redresses injuries that result from the failure of one party to live up to his part of a prior agreement. So, unless Acme has bargained with Widgets, Inc. to cover the damages that might result from an unknown intruder using Widgets' computers to launch an attack against Acme's systems, contract law would not provide relief. The rule in most cases is that the company used as a cut-out will have no prior relation with the damaged party, thus eliminating any hope for redress under contract law.

As a result, victims are likely to seek compensation for their losses by resorting to the civil arena, where the actual perpetrator need not be identified, the pool of entities with the ability to redress losses is much less discriminate (read: deep pockets), and prior promises need not exist. Specifically, negligence claims may be potent if the victim can show that the "insecure" party (1) owed a duty to use reasonable care in securing its computer systems; (2) breached that duty by failing to maintain adequate computer network security; and (3) was a reasonably recognizable cause of actual damages that resulted from his insecure computer network.

If we agree that there should be a standard of care to secure networked computers, thereby favoring a legal right to recover from the "insecure" party in negligence, we must ask:

What is the basis for imposing a duty to secure its computer system?

Who does that duty apply to in the Internet community – SWV, WebCo, ISP, and/or user? What is the scope/standard of care for each party?

Basis for Imposing a Duty to Secure Computers

Although foreseeability of harm is a primary determinant in deciding whether to assign duty, factors such as competing socioeconomic policies, assumption of responsibility by the allegedly negligent party, and the injured party's reliance have been instrumental.

FORESEEABILITY OF HARM

To say that WebCo had a duty to protect Jill User from harm as a result of a computer intrusion means that there is a standard of conduct (see Fig. 1) that WebCo must fol-

low for the protection of others on the network against unreasonable risks. Namely, WebCo must use reasonable care in adequately securing its computer systems.

STANDARD OF CONDUCT

Reasonable Care – what reasonable measures can be taken to secure your system?

Foreseeability – if those measures were not taken, who would be harmed?

Reasonably Perceived Risk of Harm – was the harmed party created by not reasonably securing your system?

Figure 1

What is "reasonable care"? It is the attention, knowledge, intelligence, and judgment defined by society for its protection. These objective qualities have traditionally been measured by the foreseeability of injury to the aggrieved party.[1] In other words, there is no duty of care owed to an injured party who is not within the foreseeable risk of harm created by the defendant.

SOCIOECONOMIC POLICIES

The costs associated with insecure computers on the Internet weigh heavily in favor of assigning a duty to secure systems. Direct monetary damage due to denial of service (DoS) attacks and unauthorized compromises can, have been, and will continue to be substantial. This can take the form of business downtime which is often measured in terms of revenue losses, compensatory payments, employee downtime, inventory costs, depreciation of capital, and actual damage to a company's own computer systems. For example, distributed network sites can lose $20,000–$80,000/hour in centralized network downtime.[2]

Other indirect monetary costs take the form of security infrastructure upgrades, loss of customer base, damage to business reputation and public image, destruction of potential partnerships, delays to market, and capitalization losses. For instance, the infamous February 2000 DDoS was estimated to have caused about $1 billion in capitalization losses and $100 million in lost sales and advertising.[3]

Duty creates an incentive to use higher care. If parties are not held accountable by liability for failure to secure, there is an economic incentive to use the lowest care. For example, if there were no law against theft, would people think twice about taking without paying? Would spammers continue to disseminate digital junk mail if they faced stiff fines?

The need to secure information will persist and magnify. For instance, in the corporate world where intellectual property is often the only thing separating competitors, it is cheaper and easier to steal information than to develop it.[4] The motivation driving acts of theft, destruction, and misfeasance combined with increasing expertise, sophistication, and effectiveness of attacks on networked computers ensures the importance of information security.[5]

Security will become more difficult and important as evolving networks grow increasingly complex. This complexity means that security bugs in software will proliferate, vulnerabilities will multiply with the increased modularity of software, extensive testing will be demanded, and security analysis will become more difficult.[6]

This also means that the danger of false victimization claims grows more prevalent in complex digital environments. For instance, the added functionality that is in the fore-

front of system design comes with a vulnerability cost. The appliances and other devices being made with programmable computer chips and Internet access are a case in point. Technology will advance regardless, but without assigning due care, there should be little expectation of security. But by assigning reasonable security measures, the wildfires due to insecurity can be downgraded to a controlled burn.

Finally, assigning vendors, service providers, WebCos, and users a duty to secure distributes the risk of loss among the people who employ the technology. This policy recognizes that no single entity is responsible for the security of the entire Internet, but each should be responsible for his/her identifiable part.

. . .no single entity is responsible for the security of the entire Internet, but each should be responsible for his/her identifiable part.

REASONABLE EXPECTATION OF SECURITY

Reliance by Injured Party

In general, reliance on the performance of another party can factor into the imposition of duty. If A depends on the protection of B, and B has knowledge (actual or imputed) of that dependence and the ability to protect, A is relying on the security capabilities of B. If you give your credit card number to a WebCo over the Internet, you depend on WebCo to protect this fiduciary data, WebCo is aware that this number is not for public distribution, and it has the ability to safeguard this data. In this way, you have a reasonable expectation that this information will be kept secure and rely on WebCo to implement appropriate safeguards. However, WebCo may not be using reasonable care if your credit card number is stolen from its database because it was stored unencrypted on the Web server.

Reasonable expectations of security are created in various ways and help determine who is entitled to protection. Industry customs are one way to measure the objective reasonableness of a victim's expectations of care. Widely disseminated bulletins (SANS, CERT/CC, BugTraq, etc.) and company policies and procedures that address computer security put users on notice that there are generally agreed upon methods to assess security and protect systems. This notifies people that data and transactions, for example, can be secured, and their subsequent actions are made with that in mind. In this way, duty may arise from information security best practices that shape the expectations of people outside.

Reasonable expectations of security are also shaped by the discrepancy in authority between the injured party and the allegedly "insecure" party. Because of the authority and control exerted by landlords over common-use areas (walkways, stairways, elevators, lobbies, front doors), they may have a duty to secure and can be held liable when defective security exists. Likewise, there are entities that have the ability and authority to manage the risks of network insecurity. When this is manifest, that party creates a reasonable expectation that security exists and will be maintained.

For example, an ISP which exists to provide users with Internet connectivity, could reasonably be relied on to forestall or mitigate the damages from a DDoS. The reciprocal knowledge that ISPs can monitor and control network traffic affecting users may create a reasonable expectation that the ISP configures routers to block directed broadcast traffic during a DDoS, for example. The ISP is aware that users are cognizant of this attack yet are incapable of implementing the same level of protection. As such, the ISP's knowledge of users' reliance on its authority to implement security may provide a basis for imposing duty.

Each party who affects computer network security . . . may owe a duty to exercise reasonable care in maintaining adequate computer security.

Assumption of Duty by "Insecure" Party

Another basis for imposing a duty to secure may arise when one party assumes the responsibility and places the injured party in a worse position. This is similar to duty based on reliance, but involves more explicit assurances by the "insecure" party. Here, reasonable expectations of security may arise when one party makes representations as to current/future security assurances or voluntarily assumes control of security, and leaves another party in a worse position by failing to use reasonable care.

A party who voluntarily assumes the performance of a duty is required to do what an ordinary, prudent person would do in accomplishing the task. If a landlord installs an alarm system leading his tenants to forego deadlocks, and an intruder causes injury because of careless installation, the landlord may be in dereliction of duty. Likewise, a software vendor may create a false sense of security in its product by misrepresenting protections or implementing them carelessly, thereby causing an end user to eschew other safeguards. In this way, the vendor has assumed the duty to disseminate a reasonably secure product and has left the user in a more vulnerable position, thus providing a basis to impose a duty to secure.

Who Owes a Duty to Secure Computer Systems?

Each party who affects computer network security – software vendors (SWVs); services providers (ISPs/ASPs); WebCos and their respective IT managers, directors, and system administrators (sysadmins); individual users – may owe a duty to exercise reasonable care in maintaining adequate computer security. The standard of care / scope of the duty will depend on the quality and quantity of the measures needed to secure relative to the actor's ability to control, assumption of responsibility, and/or socioeconomic concerns.

Software Vendor/Manufacturer Duty

Should a vendor be liable for failure to secure when its software provides the means for an intruder to damage an end user (ISP, WebCo, or consumer)?

Foreseeability: Knowledge and Ability to Control

The harm to users of software with known vulnerabilities is foreseeable, and prevention is well understood. For example, developers of Web-server applications invariably focus on business and technical concerns (functionality and time-to-market) at the expense of security, thus allowing attackers to deviate from the script's intended application. It is no secret that programmers have had the knowledge and ability to deal with buffer overflow vulnerability for decades. Since the coding and hardware solutions slowed down the program, buffer checks were eliminated.[7] This appears to be the rule, as newer versions of products continue to harbor the same vulnerabilities that plagued earlier versions. Repeatedly condoning demonstrably flawed designs proven to be problematic is remarkable because it indicates a conscious choice to disregard security measures in the face of knowledge of their importance.

Also, the mere existence of security vulnerability alerts, posting of patches, and prerelease warnings by both the vendors/manufacturers independently and in response to security watchdog bulletins[8] show tacit knowledge that these products are routinely targeted by intruders as a means to break into systems. Notwithstanding these indicators, foreseeability of harm to victims would be imputed to vendors by virtue of the widely disseminated news of Web companies, users, and ISPs' incurring disruption in business or theft of information because of the exploitation of product vulnerabilities.

Although knowledge alone would be insufficient to impose a duty, the SWV has the ability to control the extent of many security exploits. Just as gun safety would be more easily enforced if safety locks were required of manufacturers rather than solely relying upon user adherence to a wide array of handling and storage procedures, SWVs are in a position to design-away the Achilles' heel of computer security.

. . . an operator's safe configuration of software is only as good as the underlying code.

Party in the Best Position: Burden of Security

Another factor used to determine whether SWVs owe a duty to help secure networks looks to the party in the best position to secure networks. This may be judged by the relative burden of implementing security along with any negative social consequences. The technical burden involved with security evaluations of complex systems weighs in favor of SWVs bearing the brunt of implementing security in product design. In addition to technical imbalance, quantitatively it is more reasonable to assign software security to a single body of producers versus shouldering it on the product's 100 million users, for example.[9]

There is a drastic imbalance between the knowledge and skill needed by ordinary users to install and operate programs versus the technical proficiency and resources needed to configure and run them securely. This holds true for system administrators, albeit to a lesser extent, insofar as the skill and resources needed to secure systems are far more demanding compared to the ease with which harm can be wrought in this automated attack environment.[10]

Further, the technical proficiency expectations of IT professionals are irrelevant if the vendor produces a digital land mine. Just as a contractor can follow a blueprint copiously yet construct a house that crumbles during inclement weather, an operator's safe configuration of software is only as good as the underlying code. Thus, reasonable preventative measures imposed on SWVs – programming against known/knowable security vulnerabilities, and shipping software with safer default settings – would stopgap the source of a majority of network insecurities and further society's interest in maintaining a secure computing infrastructure.

Opponents to assigning duty on SWVs argue that doing so would unduly hamper market competitiveness by elevating operational costs, inhibiting functional improvements in software, and impeding product releases, the costs of which will ultimately be borne by the end user. However, ascribing a duty to exercise reasonable care does not entail wholesale abolition of every software vulnerability. Rather, it balances the responsibility in proportion to the level of authority. The alternative to not extending this duty to SWVs is to foster an unreasonable expectation that persons ill-equipped to configure for security will eliminate vulnerabilities. Furthermore, the functional effects of unusable software caused by insecure design are far more ruinous than making due with an application that does not auto-complete words, for example. Indeed, the expenses associated with cleaning up after an intrusion that could have been prevented by more secure software are much more prohibitive than heightened product costs at the front end.[11]

Reasonable Expectations of Security

Software end users have a reasonable expectation that the product will not be an open invitation for malicious intrusions. That is, a user who relies on the SWV to disseminate a product that functions adequately and does not place him in a worse position for having purchased it, is not acting unreasonably. For example, if a SWV makes a word processing program that bars a user from composing a simple letter, or exposes

. . . it is unreasonable to expect users to appraise the security of off-the-shelf software . . .

the user's entire system to any number of invasions, that SWV has created a plight for the user. What if a homeowner bought an air-conditioning unit that arbitrarily opened doors and windows at any time of the day or night to enhance cooling features?

One need not search far for examples of software that was bought with an expectation that it would perform as advertised, yet placed the user in a detrimental position. The MS Office Assistant feature illustrates how a vendor created reliance on the part of its customers and then left them in a worse position. Little did users know that when the jovial Paperclip prophetically appeared at the behest of a comatose user, the scripting technology that fueled his trojaned white horse enabled yet another back door into the application and system at large. When a Web page or HTML-enabled email was clicked, the script could add or delete files. The distinction with this security hole is that it was not a result of poor programming but was an intended "feature" built in to the program to allow the vendor to run macros through a back door. Even an exceptionally knowledgeable and scrupulous user who may have attempted to verify the risks of using this type of scripting would have found it to be labeled "safe."[12]

What is more damning is when a SWV makes explicit security pledges that are false. Users' reasonable reliance on the security of software is betrayed when prophylactic statements are made, the vendor is aware of the confidences created, and the admonitions are false. Simply put, this is Misrepresentation, and to not hold the vendor responsible for resulting damages is to invite deception. It is akin to a landlord making assurances about apartment-complex security, yet placing a master key ring in the lobby without informing the tenants about the very existence of the keys, let alone their open accessibility.

For example, labeling a control "safe for scripting" exemplifies how a relationship between parties with unequal knowledge and capability fosters a reliance that can place the "weaker" party in a worse position. Controls, such as Active X, are used extensively throughout Windows platforms, especially in Web-based applications. Safety assertions in this context can reasonably be interpreted to mean that the control cannot be used by an intruder to damage or compromise one's system. Yet, auditing or examining control properties are arduous and invoke the use of a specialized tool within the Windows registry. Coupled with the fact that controls are ubiquitous, it is unreasonable for users to discount the patent safety assurances of a product licensed by a dominant software manufacturer, and have the ability and wherewithal to search and verify the veracity of such avowals. Thus, a high degree of trust must be placed in the vendor-author that when viewing a Web page, newsgroup posting, or email message containing the safety-branded control, an intruder will be prevented from disabling Office macro warnings and executing arbitrary code.

If it is unreasonable to expect users to appraise the security of off-the-shelf software, then absurdity transudes new meaning if vendors are permitted to issue programs that lead users to believe that an application is secure yet wreaks havoc on a system, leaving users with neither warning nor recourse. If that is the case, society should tolerate clothes irons that may discharge electrical sparks and issue warning alarms after homes are incinerated.

Socioeconomics: The Emperor Has No Clothes

Consumers are quick to demand cars free of any type of defect, yet continually accept software products that are "recalled" for being unsafe. If Ford released a car into the marketplace that was continuously being recalled for potentially injurious defects or,

rather than undertaking safety R&D, used its consumers as a crash test base for design flaws, history has shown that this would not be tolerated.

Socioeconomic considerations support the imposition of a duty to secure on SWVs. Accepting the argument that bugs in computer systems and software are inevitable, it would be unreasonable to expect that SWVs should test for and eradicate every insecurity. Nevertheless, if the current lack of accountability persists, end users will continue to bear the risk and cost associated with applying the vendors' bandaids to the broken bones that hackers can readily x-ray.

Furthermore, imposing a duty creates an economic incentive to render higher-quality products. Without liability for insecurity, there is an economic incentive to create lowest quality.[13] Therefore, unless SWVs are held accountable for designing insecure software, speed, features, and options will dominate the SWV agenda. Currently, users face an uphill battle in attempting to prove that a vendor was negligent in not using reasonable care to design with security in mind. In a dispute between a user and vendor, it is assumed that SWVs are not negligent. Since the burden of proof is on the user, the SWV is in the clear unless the user can prove the elements of negligence. Statutes such as UCITA, shrink-wrap licenses, and general disclaimers work against any attempt to prove that a SWV did not use reasonable care, not to mention the cost involved in proving this on a case-by-case basis acts as a disincentive to take on Goliath. Therefore, if duty is not defined and imposed at some point, it may be infeasible for a user damaged from an insecure software product to seek redress. Furthermore, society will grow increasingly desensitized to the real damages being wrought, and ramifications of insecure software will become an accepted cost and defining attribute of networked society.

Without liability for insecurity, there is an economic incentive to create lowest quality.

SERVICE PROVIDER (ISP/ASP) DUTY

Should an ISP be held liable for failing to implement reasonable security measures that would have prevented or mitigated damages to its customers by malicious intruders? The nature of a service provider's authority (knowledge and ability to control security vulnerabilities) and its assumption of responsibility create a reasonable expectation that it implement security measures.

Foreseeability – Knowledge of Harm

ISPs arguably possess the same awareness of intrusion methods and targeted victims as product vendors/manufacturers. Just as software vulnerabilities are a common target, so, too, are poorly configured network servers. These servers are well-known in the hacker lore and finite in number. That is not to say that ISPs ought to be Reserve White Hats, but they should have imputed knowledge of the reasonably perceived risk that an intruder will try to use their network to harm their customer(s). The reasonably perceived risk of not implementing security measures at the service-provider level is that its customers will be targeted, invaded, and ultimately damaged by online miscreants. The injured customer(s) is owed a duty since she falls within the risk of harm controllable by the ISP.

This imputed foreseeability might also extend to downstream victims of an attack launched from ISP clients. In this way, the ISP resembles a public contractor that undertakes to work in a public way such as on a highway, street, or sidewalk. Here, the ISP contracts through a service level agreement (SLA) to work in the Internet, a public way. The elevator contractor or auto repairperson is deemed to automatically foresee that negligent performance (misfeasance) will likely cause injury. A contractor's failure

Service providers are increasing their exposure to negligence liability by implicitly and explicitly assuming the duty to secure their networked customers.

to perform may lead to liability if it is foreseeable that nonperformance will likely cause injury. Likewise, an ISP's failure to implement some security within its network does not limit exposure to potential harm to its customers, alone. Other entities share and utilize the digital accessways such that even though they are not in privity of contract with the particular ISP, they are foreseeable victims of its nonfeasance.

Control

As a gatekeeping authority, ISPs are in control of their respective networks to the extent that they are the only actors capable of directly implementing security mechanisms that affect the whole of their customer base. For example, they can turn off Web connections that do not follow up with valid HTTP requests, employ tools to scan systemwide for the installation of any host or broadcaster software, and help customers prevent spammers from spoofing their addresses. The same identifying features (i.e., defined signatures) that allow ISPs to block spam are present in email viruses and empower ISPs to stop them at the email server. Nevertheless, some ISPs are reluctant to effectuate their ability to secure their networks. Similar to the SWVs' failure to code against known bugs out of concern for market deadlines, ISPs may forego filtering or scanning out of concern for degradation of network performance and additional costs.

This network control and ability to enact security therein is a unilateral capability, as service subscribers lack both the technical know-how and/or operational capabilities to implement these same large-scale security measures. For example, some viruses can only be detected and halted using server and proxy-based antivirus and filtering tools. A user's desktop antivirus product would be ineffectual. Because of this authority, ISPs should bear a duty to secure since they are capable of providing reasonable security to protect another party whose ability to provide for its own safety is restricted.

Assumption of Duty

Service providers are increasing their exposure to negligence liability by implicitly and explicitly assuming the duty to secure their networked customers. As ISPs evolve and take on more functional authority they may be ultimately self-imposing a duty to establish and maintain security. By assuming the duty to secure they may be creating a reasonable reliance that users will be protected from intrusions.

For example, some providers offer free spam blocking in response to customer complaints. This same capability and authority that enables email and Web-surfing monitoring can and is marshaled by some ISPs to prevent viruses or stopgap DoS attacks against customers on their network. This drive to satisfy customers may unwittingly raise the expectations of customers that they will be protected from common threats. Even though most ISPs are not explicitly contracting security into their service level agreements, the generic disclosure statements highlighting the company's commitment to security could bear on expectations. Take the case where a business is shut down as a result of its ISP's failure to use spoof filters, even though most other ISPs, including its competitors, successfully averted the attack. The damaged company might argue that its ISP's actions fell below the standard of care referenced in the disclosure statement and manifest by the actions of the majority of other ISPs.

Thus, providers may be opening themselves to negligence claims if customers are aware of these measures, providers realize the users' reliance upon these security measures, they miscarry these self-protection strategies, and an intruder wreaks damage.

Socioeconomics

Reliability of Web hosting services is key to e-commerce proliferation. Reliability presumes security insofar as a network service or applications with widespread and exploitable vulnerabilities cannot be counted on to deliver consistent and repeatable performance. In this way, e-commerce depends upon the assurance of secure networks. To exemplify, an insecure Web server that is vulnerable to malicious acts opens WebCos to a deluge of operational damages, not to mention the costs of reimbursing their own customers who relied upon service. This has an overall negative effect on the propagation of business and transactions in the digital realm.

A related concern raised by imposing duty on service providers is the effect it will have on business enterprise technology and the need for government regulation. As software continues to migrate from ownership of applications that are run at the user level to leasing of software maintained at a centralized network, the acceptance of these enterprises will depend on the reliability of the computing, which boils down to the security of the application.[14]

WebCo Duty: Security Obligation to Downstream Victims

Should a WebCo be liable when its insecure computer(s) was used by an intruder to damage a third party? Is it reasonable to expect companies hosting Web sites to anticipate misconduct directed at their systems? As with vendors and ISPs, WebCos should owe a duty to exercise reasonable care in maintaining adequate computer security based on the legal rationale underlying negligence. Namely, to the extent that a WebCo has knowledge and the ability to control the harm to a third party from an intruder, the situation is no different than in the physical world and the same standards of conduct should apply.

Should a WebCo be liable when its insecure computer(s) was used by an intruder to damage a third party?

To date, no US case has squarely addressed liability for failure to secure, though a presage to this novel claim arose in late 1999. Pacific Bell was the target of a class action lawsuit alleging, among other things, negligence for inadequately protecting its customers against unauthorized Internet intrusions and failure to inform them that the Digital Subscriber Line (DSL) connections were not secure.[15] Regardless of the outcome, this claim illustrates the evolution of users' expectation of care regarding ISPs' duty to secure the network. Furthermore, the decision to pursue litigation for breach of this alleged duty has lowered the threshold beyond which a multitude of similarly situated parties had not previously sought redress.

Interestingly, a case embracing this issue has been levied in the UK against a prominent American company for lax security in "allowing itself to be hacked."[16] In June 2000, a UK-based ISP sought damages from Nike.com for negligent security when its domain was hijacked. The argument alleged that by selecting the lowest form of security (called "mail-from") when it registered its site, a criminal was able to spoof email, alter Nike's registry data, and re-direct Nike.com's traffic through the UK Web server. Damages were sought for the time and money associated with administering the overloaded servers.[17] Others argued that responsibility belongs with the domain registrar, Network Solutions, for allowing the spoofed email from the Nike authorized contact without the password required to change the Nike domain status. Despite the disparate damages allegedly caused by the insecure parties – Pac Bell, Nike, or Network Solutions — the underlying thread is a demand to recognize and account for exposures created by insecure network security.

Would imposing a duty to secure its computer systems and subsequently holding a WebCo liable for failure to safeguard be an unreasonable burden?

Defining Reasonable Care in the Environet

Traditionally, downstream liability determinations hinged on proof of causation: how far down a chain of connected events leading to the injury would society be willing to ascribe responsibility? The environet (Internet environment) challenges the very meaning of "downstream" since everyone online is but one click away, placing all connected users within a reasonably perceived risk.

In other words, the pool of foreseeable plaintiffs in the physical world is limited by time, location, and predictable relationships. On the Internet, when those measuring sticks are removed, the liability chain transforms into a cloud encompassing a torrent of probable plaintiffs. For instance, in the property-based world, courts would have no trouble finding a chink in the chain of causation when One-Armed Jack sues Acme for leaving its warehouse unattended and unlocked, with the keys in the ignition of its delivery trucks. Acme's nonfeasance enabled Snidely Whiplash to abscond with the vehicle. In the midst of this transgression, he displaced Jack's limb as he was exiting his car. The same scenario played out in the Environet entails "r00t Whiplash" routing his activity through 15 different hosts in five countries and storing his exploit on an insecure host at Acme. This program directs a malicious payload at some business one week later, but Victim.com happens to suffer a denial of service (DoS) and business disruption in the course of routing the scripted traffic.

In the first instance, society is not willing to impose a duty on vehicle owners (Acme) to protect persons on the highway from thieves. In other words, since it is not reasonable for Acme to foresee that a thief would be an incompetent driver, Acme could not be a cause of the injury.[18] Applying this rationale to the second scenario, the same decision may be trivially apparent given that the harm occurred well after the insecure incident, in a location far away. However, the critical question is whether the conduct of r00t Whiplash was foreseeable. A strong argument can be made that Acme had substantial reason to foresee that maintaining an insecure site increased the risk of a compromise to its own system, and correspondingly, that a criminal would maximize the vulnerability to exact harm on others connected to the Internet. Thus, Acme would have a duty to persons on the digital highway to use reasonable care to keep its system from being controlled by a digital vandal. Under this reasoning, the chain of causation may indeed link Acme's failure to secure with the damage to Victim.com.

Socioeconomics

Would imposing a duty to secure its computer systems and subsequently holding a WebCo liable for failure to safeguard be an unreasonable burden?

When the cost of accidents is less than the cost of prevention, a rational, profit-maximizing enterprise will pay civil judgments to accident victims rather than bear the larger cost of avoiding liability.[19] Following this rationale, WebCos may choose to risk paying judgments to downstream victims injured by its lack of security or insure against the risk. This would likely mean that the insurance costs would be factored into its pricing or business costs in some way, which could ultimately translate into computer intrusion costs being borne by the parties entitled to protection.

At the opposite end of the duty spectrum, where downstream victims of insecure computers are not extended protection under negligence law, the situation resembles a digital caveat emptor. Only, in this case, it would be "let the Netizens beware," and in the absence of some contract-based theory of liability or yet-to-be-established regulation, entities connected to the Internet would assume the risk that a miscreant could attack,

intrude, and wreak damage upon them. However, history has proven that whenever a major technology or industry has proliferated to effect society at large, some measure of social control will follow. If judicial imposition of duty and liability is not one such mechanism, regulations, legislation, and insurance will unquestionably rule. One needs only to refer to the automobile industry as an illustration of how its socioeconomic impact was dealt with on all three fronts.

Foreseeability of Harm

It is reasonable to expect that companies hosting Web sites should anticipate misconduct in the form of attempted intrusions. WebCos' indifference toward the security of their machines can contribute to a disastrous loss for many other Internetizens and dot-coms. The ability to capitalize on security vulnerabilities and thereby commit crimes anonymously and more easily is what fuels the criminal element in a network society. A significant underlying theme is that regardless of the measures not taken to protect its own proprietary data or information assets, a WebCo's lack of computer security plays an identifiable part in the probability and reality of another Internet entity being intruded on and damaged. Thus, both the victim (other Netizens/WebCos) and harm (theft of information; denial of service; theft of service; damage to computer systems, etc.) are not so inconceivable as to remove them from the realm of foreseeability.

It is reasonable to expect that companies hosting Web sites should anticipate misconduct in the form of attempted intrusions.

For example, the Oregon State University computer used by the hacker claiming to steal 300,000 credit cards from CD Universe was only partially secured because it was not thought to harbor anything of value.[20] This rationale undoubtedly accounted for the slapdash security on many servers nationwide that helped make possible the infamous February 2000 DDoS attack on Yahoo, ZDNet, eBay, CNN, Amazon, and eTrade. The popular media is satiated with instances where businesses are compromised. These reports focus on the victimized businesses and efforts to trace the miscreants. What is rarely reported, however, is the trail of insecure hosts along the way that facilitated the intrusion.

In the non-digital world, a grocery store owes a duty of care to secure against vandals preying on would-be patrons. If the store owner fails to attend to security concerns – hiring a security guard, putting lighting in the parking lot, installing cameras – and someone is victimized by Hamburglar as a result, a lawsuit would be filed before the purse handle was cut. However, if Trinkets-R-Us sets up a Web server out of the box, without configuring for security, the inferential leap to hold it accountable for ensuing damages is not being made. In this case, Magic8.com may suffer loss of business for hours or days as a result the DDoS servant launched from Trinkets' compromised server.

Similarly, a company that sets up shop in the Internet is presumed to invite/entice visitors. This undertaking should be accompanied by a corresponding degree of care measured in terms of some modicum of security against third-party malfeasance. Surely, the presence of a physical threat in the grocery instance would justify a heightened expectation to secure on the part of the store. However, does the economic/physical harm distinction justify tolerating a WebCo that ignores security? This is answered by recognizing the liability realities of companies failing to protect financial data needed to consummate online purchases. If online companies take no measures to protect their customer credit card databases, thereby putting the economic health of their customer in jeopardy, courts would have no trouble holding them responsible, and the MasterCards of the world would not be so quick to write off this type of fraud.

The traditional notion that there is no duty to protect others is challenged by the ubiquitous, distributed, and tightly knit nature of network computing.

User Duty

Does an individual user owe a duty to reasonably safeguard their systems against unauthorized access for the protection of downstream victims? The answer depends in part on users' ability to implement safeguards and overcome common network vulnerabilities; users' knowledge of the risk of failing to secure; and society's willingness to expand the scope of foreseeable plaintiffs.

Control

As there is advancement in solving the issue of uniformly and reliably informing and enabling average users how to fix their vulnerable systems, control will factor into assessing a user's duty. To the extent that users have the ability to secure – through reasonable instructions accompanying a product, disseminated by a service provider, or via widely publicized bulletin(s) – their failure to install available patches or enable antivirus software may no longer suffice as an excuse.

Foreseeable Risks and Victims

As discussed previously, the computing community of vendors, service providers, and Web companies has knowledge that miscreants seek unauthorized entry. To the extent that the general user can be imputed with awareness of this penchant, he should be on notice of the potential harm in failing to secure. As network computing has become part of the daily lives of society on the whole, security issues are no longer confined to the computer-related workforce. Rather, first-hand exposure and media attention paid to security exploits has raised the public awareness and contributed to a more informed user populace. To be sure, virus propagation and malicious exploits occurred prior to the Love Bug and Trinoo, but consistent coverage in major news media headlines was unprecedented. In this way, there is a stronger argument for imputing knowledge of the foreseeable risks to end users today than even a few years ago.

In addition to the foreseeability of the danger, the conceivable parties within the purview of that danger must factor into a duty analysis for users. To a certain extent, the same popular media mechanisms (CNN Headline News, *New York Times*, etc.) that raise awareness of insecurity risks impart knowledge of the person(s) likely to be harmed. Furthermore, the ignorance card may carry less weight in situations where a user had been intruded on or infected in the past and was put on notice. Indeed, other parties connected to the network are foreseeable victims of a user's failure to safeguard his system. A user's indifference toward the security of his system can contribute to a disastrous loss for many other Netizens and dot-coms.

Socioeconomics

The degree to which courts are willing to extend the pool of foreseeable victims to encompass other networked users will hinge on socioeconomic policies. The traditional notion that there is no duty to protect others is challenged by the ubiquitous, distributed, and tightly knit nature of network computing. Based on the principle that an orderly society demands authority be accompanied by responsibility, the fact that nearly every host connected to the Internet exacts some control over the others should imbue networked users with comparable responsibility. Unlike point-to-point telephony, Internet communications are three-dimensional, thus rendering every connected host a potential portal to any and all others. This interconnectedness makes security an embedded dynamic such that there is no clear boundary separating entities on the Internet. In the property-based world, security is based on drawing lines that separate you from outsiders. If you fail to protect yourself by not locking your door, for exam-

ple, then your property is the casualty. Responsibility for the invasion does not shift to your neighbor four doors down. Correspondingly, you have no duty to secure your premises for the protection of your neighbor. Short of disconnecting from the network, self-protection in networked society is a misnomer since each host's security is linked to each other's.

Insofar as electronic commerce is driving the influx of users onto the network, placing a duty on those that choose to engage in this activity recognizes that security is an embedded risk that should be distributed accordingly. Where suppliers offer the means to prevent viruses, dissuade port scanners, or detect unauthorized access attempts, for instance, users should be held to a reasonable standard in preventing the ill-effects of these activities. For example, a user employing one or more of these prophylactics may prevent a hacker from being employed in a distributed denial of service attack against a commercial Web site.

CONCLUSION

Whether you are a victim-symptom or a liable-cause, reality dictates that networked computers are vulnerable to undesired actions and resulting harm owing to computer security vulnerabilities. The nature of this environment both forces society to redefine what is reasonable and facilitates the shifting of responsibility. To this end, the critical question to be addressed is: should insecure computers be tolerated given the nature of modern computing infrastructure? Legal claims based in negligence prove to be a viable answer, insofar as they attempt to assess responsibilities, define duties, and assign liabilities amidst this new interconnected environment, where both traditional and unprecedented relationships, conduct, and consequences intertwine.

The potential onslaught of claims arising from insecure computer systems is not a veiled threat but, more aptly, a ripening promise. This article has highlighted the parties and common scenarios likely to spawn litigation for failure to secure computer systems, between vendors, service providers, Web businesses and individual end users within networked society. Indeed, there is support for the imposition of duty to safeguard networked computers. This duty arises from traditional factors used to judge negligence: the foreseeability of harm for failure to secure; reliance on the party in the best position to implement and maintain security; the assumption of responsibility to secure; and, various socioeconomic considerations. The scope of this duty can be defined in light of these factors, recognizing the various levels of knowledge, control, and identifiable effect that each respective party has on securing networked computers against injurious damages.

References

1. *Palsgraf v. Long Isl RR Co.*, 162 N.E. 99 (N.Y. 1928).

2. The Gartner Group estimates the average cost of downtime in brokerage operations at $6.5 million/hour; eBay paid $3.9 million in credits to customers for a 22-hour service outage in June 1999. See "Towards a Dependable Self-Healing Internet," Testimony to the Senate's Subcommittee on Communications, March 8, 2000 (prepared statement of Raj Reddy, co-chair, President's Information Technology Advisory Committee; Herbert A. Simon, professor of computer science and robotics, Carnegie Mellon University). See generally "E-Business Survival during Denial of Service Tornadoes" (viewed August 12, 2000) *http://gartner6.gartnerWeb.com/public/static/store/networking.html.*

. . .should insecure computers be tolerated given the nature of modern computing infrastructure?

3. Matthew Kovar, "$1.2 Billion Impact Seen as a Result of Recent Attacks Launched by Internet Hackers," Yankee Group, February 14, 2000, *http://www.yankeegroup.com/Webfolder/yg21a.nsf/pusharea/$1.2+Billion+Impact+Seen+as+a+Result+of+Recent+Attacks+Launched+by+Internet+Hackers*. See generally Cahners In-Stat Group (viewed July 16, 2000) *www.instat.com/abstracts/ia/1999/is9906spabs.htm*.

4. See generally "Annual Computer Crime and Security Survey," Computer Security Institute and Federal Bureau of Investigation (1999) (e.g., there was a $50,000 "bounty" placed on notebooks belonging to any executive of an energy company involved in bidding on international projects). See Susan Breidenbach, "How Secure Are You?" *Information Week*, August 21, 2000, p. 74.

5. See generally "CERT/CC Overview Incident and Vulnerability Trends" (viewed August 20, 2000) *http://www.cert.org/present/cert-overview-trends/tsld001.htm*.

6. Bruce Schneier, *Crypto-Gram*, March 15, 2000, *http://www.counterpane.com/crypto-gram-0003.html*. Complex systems must be broken into manageable pieces; security often fails where two modules interact; complex systems demand increased testing of specifications, design, and implementation.

7. Matt Bishop, "UNIX Security: Security in Programming," SANS '96, Washington, DC, May 1996. "A small number of flaws in software programs are responsible for the vast majority of successful Internet attacks because attackers don't like to do extra work. They exploit the best known flaws with the most effective and widely available attack tools. And they count on organizations not fixing the problems. System administrators report that they have not corrected these flaws because they don't know which of over 500 potential problems are the ones that are most dangerous, and they are too busy to correct them all." Quoting Alan Paller, "The Ten Most Critical Internet Security Threats," SANS Security Alert, May 2000, pp. 1, 2.

8. See generally CERT/CC – Computer Emergency Response Team / Coordination Center, *http://www.cert.org*; Security Alert for Enterprise Resources, h*ttp://www.safermag.com*; Security Focus, *http://www.securityfocus.com*; BugTraq, *http://www.bugtraq.securepoint.com*; SANS Security Digest Services, *http://www.sans.org/newlook/digests/SAC.htm*; Attrition, *http://www.attrition.org*; Microsoft Technical Updates, Microsoft Security Bulletins, *http://www.microsoft.com/technet/security*.

9. For example, the Apache Web Server constitutes nearly two-thirds of installed Web servers, yet every copy is shipped with CGI vulnerabilities that can lead to root access to the server. Breidenbach, *How Secure Are You?* p. 74. There are about 100 million MS Office customers. Security Wire Digest, *ICSA Information Security Magazine*, July 24, 2000 *http://www.infosecuritymag.com/securitywire/index.html*.

10. For example, many attacks are batch mode processes that discover vulnerabilities, compromise these weaknesses, install daemons, and cover their tracks.

11. See generally "1999 CSI/FBI Computer Crime and Security Survey," *Computer Security Issues & Trends*, Winter 1999. This survey tallied $266 million in total losses due to computer security threats.

12. See, CERT Advisory CA-2000-07, "Microsoft Office 2000 UA ActiveX Control Incorrectly Marked 'Safe for Scripting,'" May 24, 2000, *http://www.cert.org/advisories/CA-2000-07.html*.

13. Bruce Schneier, *Crypto-Gram,* April 2000, *http://www.counterpane.com/crypto-gram-0004.html.*

14. "Application Service Providers: Are They Sitting Ducks?"; SQL *Server Magazine*, April 7, 2000, *http://packetstorm.securify.com/mag/winsd/winsd.040500.txt*.

15. Todd Spangler, "Home Is Where the Hack Is," ZDNet News, April 10, 2000, *http://www.zdnet.com/zdnn/stories/news/0,4586,2524160,00.html* (Nathan Hoffman initiated this suit after learning that his DSL-connected computer was wide-open to potential attacks, and discovering that he was being port-scanned many times a day from worldwide locations).

16. David Raikow, *New Legal Storm on Net Horizon*, ZDNet News, July 4, 2000, *http://www.zdnet.com/zdnn/stories/comment/0,5859,2597881,00.html.*

17. Craig Bicknell, "Whom To Sue For Nike.com Hack," Wired News, June 29, 2000, *http://www.wirednews.com/news/politics/0,1283,37286,00.html.*

18. See *Avis Rent A Car System, Inc. v. Superior Court*, 12 Cal. App. 4th 221 (1993).

19. *See* Richard A. Posner, "A Theory of Negligence," 1 *J. LEGAL STUD.* 29 (1972) (citing *US v Carroll Towing Co.*, 159 F.2d 169 (1947)).

20. Ted Bridis, "Hacker Victims or Unwitting Accomplices," Associated Press, February 10, 2000, *http://www.canoe.ca/TechNews0002/11_hackers.html.*

consulting reflections

Why I Consult, How You Can, and a Few Notes from the Field, Part 1

by Strata R. Chalup

President, VirtualNet; Starting as a Unisys 68K admin in 1983, Strata Chalup is now an IT project manager but allegedly has retained human qualities. Her mixed home network (Linux, Solaris, Windows) provides endless opportunties to stay current with hands-on tech.

strata@virtual.net

A recent offer of an employee position, doing something that would have been very interesting, made me stop and re-examine my work history and patterns of engagement. This was one of those dream jobs, where everyone says "You'd be nuts not to take this." The company was right, the team was right, the compensation was right, yet I really didn't want to take the job. In an effort to figure out just *why*, I dug through several pages of resume and came up with a startling answer: "because I'm a career consultant." Oh. Yes, I guess I am, after all.

I'm coming up on 19 years of work history since I left college and got my first IT/IS job in March of 1983. Over those years, I've spent 5.5 years as an employee at a fixed annual salary and 11.5 years as a contractor at an hourly rate. Only 21 months of my most recent decade of work have been spent as a salaried employee. As I put it to a friend, "I looked around and discovered that I had an actual career, rather than merely a collection of extremely efficient work-avoidance habits."

After a Decade or Two, Why Write About it Now?

Good point – it's mostly that I hadn't realized that it was unusual. Folks who know me know that I am incredibly well-informed on a number of diverse and eclectic topics, and for everything else I tend to live under a rock and poke my head out every few years. Consider this a blink in the sun.

I've also had a number of friends interested in exploring a transition to consulting. This article was partially composed as several long emails to friends who asked me about becoming a consultant. I also have a collection of articles, many of which have appeared in past issues of *;login:*, which I forward to them, but they still come back with questions. I hope that this article represents another trove of useful information, but you the reader will have to be the judge of that.

I highly recommend reading the excellent articles in the bibliography at the end of this article. In many cases, I feel that the authors have given such a thorough treatment of an issue that I simply refer to the article rather than attempting to restate the point. Most of them are available on the Web in the SAGE members area, and I reproduce the URLs in the bibliography section at the end of this article.

The obligatory caveat: I am not a lawyer, an MBA, or a tax advisor. Please get professional advice in the appropriate field before making any important decisions based on the information in this article. That said, I have tried to be as accurate as possible. Please feel free to email me with corrections, comments, and suggestions, and I will post them on my Web site with the original article. The narratives about forms, rules, and taxes here are unfortunately quite specific to the USA, as I have not yet had the experience of contracting in other countries. I would be very interested in hearing from consultants outside the USA about their contracting suggestions and experiences.

Is Consulting Your Style?

My own particular big kick in the workplace has been building something huge, say for 100K - 500K users, putting it in place, turning the key on it, and knowing it will

run smoothly to its limits. I don't enjoy watching over it afterwards. I don't enjoy planning the next level of service. I don't enjoy marketing it or leveraging it. I don't get into the "blah blah Ginger blah power blah shape industry blah Ginger" thing. Designing and/or architecting a fairly large multi-protocol product or service and leading a team to implement and/or deploy it is not something you get to do very often in most employee positions. I try to take substantial vacation time between projects, to keep from burning out. For me, contracting and taking time between gigs is an ideal work situation.

I try to take substantial vacation time between projects, to keep from burning out.

In contrast, I know quite a number of people who are happiest going deep and thorough on a particular product, solution, or technology. They want to know every aspect of it, get involved at every step of the process, and apply iterative refinements. Another group of people get most of their job satisfaction not from the technology itself, but from the interactions with people. They take pride in being part of a well-functioning team, no matter what they are doing, and feel a strong attachment to the team and the organization. Both of these types of folks could do well at consulting, but might find it difficult for them to get the same opportunities to experience what really motivates them in the workplace.

The other aspect of consulting that appeals strongly to me is that of trading autonomy for money. I've always put a high value on keeping control of my own time. For the duration of a contract, I can be on-call, I can make tight deadlines, I can practically live at the office. Entirely of my own free choice.

If I am going to be put through the wringer, I can choose to insist on compensation proportional to the trouble, or I can choose another contract. Either way, I am making a choice. I am not "being a team player" and giving up a night, a weekend, a planned vacation "just one more time to get us out of a jam." For me, that element of choice makes all the difference.

Another aspect of consulting that many people, myself included, enjoy is the ability to practice in areas of work that are not part of one's usual job duties. I have done specialized technical training seminars in the past, and am once again developing seminar and tutorial material for short topics such as IT problem-solving skills and essential project management for IS/IT staff. This is a far cry from building out network services projects, but it's something I enjoy and can do now and then when a client asks for it. It would be difficult to combine giving seminars with a traditional employee position. Similarly, I know of people who do senior systems consulting, but now and then take gigs to develop a Web presence for a small business or for individual artists. This kind of freedom to do what you really love, as well as what pays the bills on an ongoing basis, defines for me the essence of consulting.

Off to a Rough Start

It's considered conventional wisdom to "get it in writing," and all the more so when you're just starting out. You ought to have a well-defined Statement of Work or description of contract duties, as many other articles have mentioned. It can also be a very good idea to get explicit signoff on anything that seems, well, let's be polite and say "counter-intuitive."

My first client out of school in the early 80s was a little startup that wanted to do a videodisk-based arcade game. They were very concerned about security, to the extent that they would not give me the root password to the machine on which they kept all

The word on the street is often, "you have to be incorporated to be taken seriously!"

their financials and venture capital contacts. Not even for an hour or so to set up backups!

These were all Codata or similar 68000-based UNIX boxes, with local tape. I gave them line-by-line on what to put in the crontab file, copied from other machines successfully set up. I also printed out and saved all their emails promising to install the script and saying, "no, we won't give you access, we'll take care of it." I requested that one of the principals, my immediate supervisor, sign and date the printout.

A few weeks after I'd moved on, I received a panic call. They had lost the hard drive on the "special" machine. Where were the backup tapes? Sadly, there were none – they hadn't followed my instructions, and had never made any. They were completely out of luck. They threatened to sue me, until I faxed them the emails, including the signed copy showing that they had taken upon themselves the responsibility to back up that system.

I didn't have any money, so suing me was a pointless gesture, at least in terms of their recovering the data or minimizing their loss. The real goal was to transfer the blame so that their angel investors wouldn't realize how irresponsible they had been. Welcome to contracting! My next job was an employee position, but it took only a short time before I could no longer resist the siren song and took up contracting again.

To Incorporate or Not?

I strongly suggest talking to people who have incorporated and run their business for several years about some of the details. I can only speak from the non-incorporated side of the fence. That said, let me try to illuminate some of the issues for you, to help you make your choice. Some of my own choices may be more based on inertia than strategy at this point, given that I have been primarily a consultant since the early 80s, and only took the step of registering a business name in 1993.

Incorporation As a Tax Strategy

Other people's accountants tell them to go do it. The word on the street is often,"you have to be incorporated to be taken seriously!" My tax lady says its overrated, and can get people in trouble easily. What to believe? Every case is different, but here is an example that illustrates how complicated things can be.

First off, a quick definition: Schedule C, a US tax form, the "Statement of Profit or Loss from a Business." If you are a sole proprietorship or running a non-incorporated business of any kind, it is the additional form that you file with your form 1040. You are also going to file a Form SE, to calculate your Self-Employment tax, but the articles in the bibliography go into that in detail, so we'll just mention it here. There are different forms that are filed for corporate taxes.

There are two main types of corporations, traditional C corporations and the newer S corporations. The US tax codes have some positive bias toward small "family" businesses, so an individual filing a Schedule C has historically had some advantages over corporate filers. S corporations were invented to fill the gap, recognizing that there were many small or individual-run businesses which needed a corporate structure for business or liability reasons but which might, in some opinions, be entitled to a bit of a break compared to a large traditional corporation. S corps share many of the advantages of Schedule C filing, in particular the less-complicated recordkeeping and the ability of business income to "pass through" the S corp into the total annual income of

the individual. This avoids the double taxation scenario of C corporations that I will describe below.

Let's say that you C-incorporate, and in a year you make $120,000, paying yourself a salary of $10K gross per month. The corporation is taxed on any profits, and you are taxed on your income. If you don't leave any profits in the corporation, you aren't double-taxed. You are only out the time, hassle, and higher tax prep fees of dealing with corporate tax preparation and mandatory corporate reporting to whatever state you incorporated in, since you have to file corporate taxes whether or not the corporation made a profit.

You are unlikely to follow this scenario, though, since it would not allow you to deduct any expenses of running your business. OK, you can deduct them and run the corporation at a loss, but it doesn't do you any good in terms of saving money. If you do leave profits in the corp, such as by paying yourself a salary of $5K/mo in our example, those profits are taxed at the corporate rate, which is probably lower than your individual rate. You also can deduct your expenses and run at a lower tax rate overall. Let's say that you have $60K of "profits" (over your salary) and $20K of expenses. After you pay your taxes, and your overhead (social security, your medical plan, etc), say there is $35K left in the corporate account. Great! Or is it?

Here's the problem – how do you get that money out? Say you suddenly need a new roof, or your kid needs new braces, or whatever. You have to get legal/tax advice on how to "get at" that money, even if you are a limited corporation with just you and your spouse as officers. If you give yourself a loan, you may be liable for tax on the difference between the interest rate the corp charges you and what market rate would be. If you just take money out, you are in big trouble. If you pay yourself a special "bonus," you may or may not be in trouble depending on how you do it, whether you have other employees who are treated differently, whether you have set up rules on your books that allow for it, etc etc etc.

After you jump through the hoops to get to the money you'd earned earlier, your disbursement of it may end up being double-taxed, since the corporation has already paid taxes on it, and now you are going to pay taxes on it as regular income. Again, S corporations avoid this issue, but carry restrictions different from C corporations – you can't take an S corporation public, for instance! For most of us, this is not an issue, but if you are a consultant who is accumulating intellectual property or reputation in the hopes of a liquidity event someday, it may be a concern for you. There are issues involving retirement planning as well, most of them centered on rules to enforce "fairness" among employees of a corporation, even an S corp. We discuss this slightly in the Retirement Planning section later in this article.

Again I must stress that there is no substitute for doing your own research. Nolo Press is an excellent resource – consider them the O'Reilly of legal advice books. They even have a similar origin: the founders wanted to publish books about a highly technical field accessible enough to help the average layperson make some educated decisions on his or her own.

Property and Services Issues

Suppose you plan to keep some money in the corporation to buy something you need, like a new laptop, or to pay for a DSL line. For services, recall that if your corporation is paying the bill, you may be locked into higher "business" rates for phone, DSL, or whatever. If you are paying the bill in your own name and expensing it, you can get

. . . property you buy via a corporation, C or S, is not yours. It belongs to the corporation.

around that, but now you have an additional audit trail to maintain and two sets of reports to file on it, not one. If the company is paying the bill and you are not expensing it, you need to document that it is necessary for you to do your work as an employee of the company or it could be a taxable benefit to you. Doing this sort of paperwork is not optional, though many people do not do it. If they are ever audited, these folks may wish they had been documenting the issue.

If you are doing a Schedule C sole proprietorship, you have only one set of records to keep and file. On a Schedule C you may directly deduct up to $17,500 of capital equipment purchases in one fiscal year. This number varies from year to year, so be certain you know the appropriate figure for the fiscal year. This is an order of magnitude easier than keeping depreciation tables on equipment. If you are C-incorporated, you have to use the depreciation tables (last time I looked). To be fair, though, in the Schedule C case, you can only deduct the percentage of cost that corresponds to your percentage of business use, whereas if your corporation acquires the laptop, you as an employee don't have to track your business vs non-business use. S corporations may also take the one-time deduction, and thereby avoid depreciation tables.

Recall also that property you buy via a corporation, C or S, is not yours. It belongs to the corporation. If you want to sell it, give it to a friend, etc, you have to do the paper trail. If you sell your cousin Sally a laptop for $100, and that laptop is valued at $1400 on your depreciation tables, you or she may get the attention of the IRS. Even if the street value of the laptop is consistent with the price, you will have to follow the rules. If your S corporation has multiple owners, you will need to get even more complicated. Fortunately, most multiple owner S corps are usually an individual and his or her spouse, who are likely to be filing jointly anyway, thus minimizing the hassle. Talk to your tax person!

A colleague of mine mentioned that one may get around some of the residential vs business issues for services such as DSL and phone by placing the order personally, but paying with a company credit card which uses one's name (and possibly one's home address as a billing address). The sort of vendors who care whether it is a personal or business transaction are typically not in a position to see the physical card, and what they don't know may not hurt them. It is up to you to decide whether this approach works for you. If you are going to be using a resource, such as a residential phone line, at a usage level that is more like that of a business, you may feel that it is more fair to pay the difference.

Incorporation as a Differentiation Strategy

Incorporation will probably protect you better from the IRS exclusionary rule if you plan to work mostly with one client. I usually have several clients over the course of a year, so that isn't an issue with me. At this point, I also have a history of Schedule C filings going back over a decade. Get the IRS "Employee or Contractor" document at *http://ftp.fedworld.gov/pub/irs-pdf/p15a.pdf* and read it carefully. It talks about employee vs contractor characteristics and also about things like employer loans to employees and the taxable consequences thereof. A potential contract employer is likely to be very concerned as to whether you could be later be considered an employee by the IRS – or by the courts! The Microsoft ruling has made many firms skittish.

Strongly consider getting an Employer Identification Number (EIN) as a way to differentiate. If you choose incorporation (C or S), you will get one for the corporation in the process, since a corporation is a legal entity. An EIN is to an organization or busi-

ness what a Taxpayer Identification Number (TIN, formerly SSN) is to an individual. You can request one from the US Government and use it for your business. If you have more than one business, you should get an EIN for each of them. In a wonderful case of "the shoemaker's children go barefoot", I have yet to file for my own EIN, and will have corrected that by the time you read this article.

DBA: It's not Just for Databases...

If you are not incorporated, you cannot legally call yourself "Joe Admin Consulting" without registering that as a business name. There is a fairly simple set of forms available that let you select a DBA, or "doing business as" name. I went to the San Jose City Courthouse to file mine. You will need to look through the list of existing names to check conflicts. Back in 1993, this was a huge notebook with a set of monthly update inserts stuffed into drawers at the same table. I hope that things are a bit more modern now. Technically, a DBA is only valid at a state level. If you are doing business in multiple states, you may wish to secure the same DBA in other states.

You pay a modest fee to secure your DBA, and are required to give public notice of your acquisition of the name. Public notice is quite strictly defined as publication of a notice in certain qualifying newspapers deemed to have sufficient coverage in your area. In San Jose, several qualifying papers were within walking distance of the Courthouse, so it was an easy matter to arrange for the publication of the required notice. The DBA may be valid "permanently" or for some number of years, such as 10 or 15. Make sure you know when or if yours expires – it's easy to forget over such a long period of time.

You may also wish to obtain a business license from your town or municipality. In Sunnyvale, there was a modest fee, and the fee was unchanging as long as you were not having customers visit your home business location. It was presumed that if customers were visiting the site, you were doing something profitable enough that the city needed to get a cut from it, and thus your fee would be set annually based on your tax returns. It is worthwhile setting a policy of not meeting clients in your home office simply to comply with this type of requirement, as well as to minimize problems with neighbors or landlords.

Going back to our original discussion in this section, one colleague said that when he proferred an EIN and DBA (i.e, a registered business name) to a potential client instead of his own name and a TIN/SSN, "all of the 'we don't wanna do 1099' crap went away". Wonderful! Right? Yes, as long as you're prepared – read on.

System of Cheques and Balances

There is one particular downside which accompanies using any name other than your own for business, whether you are an official corporation or an individual with a DBA. That is the insistence by your bank on not cashing checks that are made out to a name that doesn't correspond with a valid account holder. You will need to open a separate business account in order to cash those checks, and you will need to present proof that you are entitled to use the name, such as notarized copies of your incorporation papers or of your official DBA notification. It can take several weeks (or more!) for the certificate of your DBA to be mailed to your address of record. If your bank of choice, like many, requires a copy of that particular document, you may have to wait weeks or months to open that account. One friend had the embarrassing situation of having to request his client to cut a replacement check for him. His busy schedule led him to wait a couple of weeks before depositing the check, only to find out that the bank

Contrary to popular belief, incorporation will *not* provide you with any substantial personal liability protection.

would not cash it, as it was in the name of his consulting business only. As with many businesses, the client had preprinted checks which indicate that the check is only valid up to 60 days after the issue date. Sixty days can pass very quickly – my friend did not get his paperwork in hand in time to open the account before the check expired! Fortunately for him, his reputation with the client was well-established through a prior business relationship with his client contacts, and he continued to work with that client.

In my case, my bank used to offer account holders the option to add a DBA signature card to a regular checking account. I chose to do this rather than open a separate account for my business, since I already had other accounts for savings and investments. I can use my individual "consumer" account to cash checks made out to either "VirtualNet" or to "Strata Rose Chalup". If I stop by the teller window at a branch that doesn't know me, they can see it on the computer record attached to my account and honor the check. The bank I use discontinued this policy sometime in the late 1990s, but as long as I don't change my account number (such as by changing home branches), I am grandfathered into the ability. If you are just starting out, or are on a very tight budget, it might be worth seeing if your credit union or similar institution might offer you the ability to add a DBA signature card instead of opening a commercial/business account. I should probably just open a separate account one of these days, but right now I'll stick to first principles – "if it's not broken, don't fix it, just plan an upgrade for later."

Incorporation as a Way to CYA

That's "cover your assets", for those who jump to conclusions. Contrary to popular belief, incorporation will not provide you with any substantial personal liability protection. The IRS and the justice system recognize that the type of small corporations (S corps, for instance) that people form for consulting businesses are often used to try to evade liability. Even traditional C corporations are finding that key officers are being indicted and prosecuted for their role in issues like tax evasion or illegal behavior. In a case of negligence, breach of contract, or other non-criminal issue, the lawyers will go after you as the majority owner/officer of the corporation. This is completely standard, and you may expect it in any of the 50 states.

In California specifically, incorporating won't even give you basic tax protection in the event of an accidental or deliberate error in tax payments or filing. It is quite common here for the state to freeze personal accounts of listed corporation officers in small closely-held corporations if there is a tax issue. They are technically not supposed to, but try getting a judgement to that effect in the California courts! Especially with a frozen bank account. My tax lady has told me some pretty scary stories, and I've seen some corroboration on the Net. I don't know if this is a problem in other states. California seems to have very lax standards in some things regarding state responsibilities to individuals and corporations – remember getting your tax refund "voucher", instead of a valid check, a few years ago?

If you do not tend to carry any business insurance of any type, you may be surprised to know that your home or renter's insurance almost always carries a rider saying that anything used as part of a home business is specifically *not* covered by your insurance. This is the case whether you have incorporated or not, and your insurance company can request a copy of your tax forms to determine if equipment used for business is included in your claim. Business insurance policies often do not cover computer equipment, or may set an unreasonably low cap on the amount which may be claimed.

If you travel with a laptop frequently, I recommend Safeware, which offers a computer policy including laptop screen replacement and full replacement value. At present, $7500 worth of coverage costs me less than $100/year.

Some companies will be fussy about doing "corp to corp" billing if you are not incorporated, but become non-fussy if you show them proof of liability insurance. Some will insist on proof of liability insurance whether you are incorporated or not. Most companies do not, and assume that you have it if you are incorporated. This is an understandable belief, because traditionally a corporation has vital assets to protect and would carry insurance. It's usually incorrect, but it's understandable.

Errors & omissions insurance can usually be had for $750 - $1500 for a 6 month policy covering $1M in damages. Details will vary depending on your work history, nature of work, and ability to figure out the incredibly Byzantine forms provided to you. It can be a good idea to keep this type of insurance at all times, or you can rely on details and disclaimers in your Statement of Work to carry you through. I have gotten liability quotes and forms from our regular insurance agent, who contracts with Farmer's for our cars and renters insurance. Try your regular insurance agent first – they will probably give you the same policy at a better rate. Farmer's and similar companies don't actually do this kind of underwriting themselves. Your agent will use his or her status as a licensed insurance agent to obtain a policy directly from firms specializing in such things. Since your agent already has your business on other policies, and has a different cost structure than your bank or an insurance firm specializing in business insurance, it is highly likely that he or she will not tack on extra fees beyond the built-in commission rates. Thus you are apt to get the same exact policy at a much better rate.

W2 or 1099? We Like Both!

W2 hourly consulting is a great tool. Don't listen to folks who tell you that you have to do everything on 1099 so that you can deduct against it. Remember that on a US Schedule C, you must show a profit at least three years out of five. Okay, if you are doing weird stuff like raising race horses or a couple of other activities, you can do five years out of seven, in which case you're unlikely to be reading this article.

I have a good idea of my deductible business expenses, since I've been tracking them on Schedule C's for a number of years. I generally have a pretty consistent level, which fluctuates upwards by $3K to $5K in years where I'm upgrading my computing infrastructure. I go to pretty much the same conferences every year – very few, since the deductible expenses don't even begin to outweigh the loss of a week's income to attend the conference. When I am planning my annual income goals, I set a goal of doing enough 1099 work to cover my expenses plus a clearly demonstrated profit that will satisfy IRS regulations.

Once I've met my expense and Schedule C profit goals for the year, then it's time to look at hourly W2 income. As a reminder, even if you are contracting hourly, if the income is W2, you can't use it to deduct against on a Schedule C: only your 1099 income is counted as your business income. Beyond that, there is no particular advantage to 1099 over W2, and there are some advantages to hourly W2.

The hourly W2 contract often comes with benefits like a 401K plan, medical insurance if you work with the contracting agency for some specified amount of time, and so on. This can include COBRA eligibility later on. Agencies can generate new business for you, serve as references, and save the day by doing pass-through billing to clients who balk at dealing with an unincorporated consultant. Recall that when you are making

Don't listen to folks who tell you that you have to do everything on 1099 so that you can deduct against it.

1099 income, you are paying your own Social Security tax. When you are on hourly W2, the agency is paying it. Since there is an upper limit on how much Social Security you pay in annually, this will save you money in the long run. Maybe you will reach your cap entirely on hourly W2, thus losing only the 7-8% SS and related tax directly rather than paying any of the 12-14% Self-Employment tax.

A quick tip: there is a "magic form" which is filed with your taxes to requisition that any excess Social Security taxes deducted from W2 wages be either refunded to you or applied to your tax bill. If you have done substantial, highly-paid W2 hourly work for more than one employer in a given tax year, it is almost certain that you are several hundred dollars over your Social Security cap. Why? Each employer will insist on calculating the deductions as if that employer were your only job in that tax year. Their accounting software is just not set up to do anything differently. A good accountant (or tax program) will notice this and include the form to be filed with your main 1040 forms. If you missed this last year, it may be too late or it may not – if the cost of re-filing is much less than the amount you could claim, it may be worthwhile. Re-filing amended versions of previous tax years is definitely one of the many cumulative audit flags with the IRS. Depending on how many other audit flags your may have in your filings (home office deduction, business use of a personal vehicle or vice versa, etc), you may find it wiser to let the money go and remember to do differently next year.

Retirement Factors

If choosing between two similar W2 opportunities, check the 401K rules. The one that lets you contribute immediately is the winner. You may not be around long enough to vest on any matching, but you always own your own contributions. Be careful if you work for multiple employers with 401Ks in a given year – they will deduct the percentage you tell them, not the absolute dollar amount. It is up to you to watch the deduction and calculate the correct percentages to deduct so that the total of all 401K deductions across employers does NOT exceed the annual cap for that year.

I am not certain of the interaction between 401K and SEP-IRA plans, since I have usually only had access to one or the other. I know that in some cases you may be able to open a regular IRA along with a SEP IRA if you have both W2 and 1099 income.

I am not a tax preparer or tax expert. Check *everything* here against a real tax preparer to make sure that some misunderstanding of mine does not create trouble for you! The IRS has a nice site optimized for small businesses and self-employed types, at *http://www.irs.gov/smallbiz/index.htm*. If you are not living in the US, much of this section will probably not be useful, though there may be analogous tax situations in your country. For specific info on retirement plans for small businesses, see IRS Publication 560, available (regrettably not as a single document) online at *http://www.irs.gov/forms_pubs/pubs/p560toc.htm*. You will probably also want to look at the FAQs for IRAs at *http://www.irs.gov/forms_pubs/pubs/p590toc.htm*. Publication 560 in particular is written for those who are paying others, rather than someone self-employed paying him or herself, so it may seem confusing. Generally you will be opening a SEP-IRA at a financial institution rather than registering your own IRS-approved SEP, which can explain some of the confusion there.

Note that if you chose to incorporate, your retirement planning also gets a bit more complicated. As long as you are the only employee of your corporation, things are a bit simpler. Retirement plans are not supposed to favor "highly-compensated" employees more than other employees, so there are complicated rules that need to be considered.

A nice glossary of the different plans is on the Web at a major brokerage site: *http://www.charles-river.com/benefits/retirement/glossary_of_terms.htm.*

An extremely nice comparison of plan types is also found there: *http://www.charles-river.com/benefits/retirement/retirement_plan_comparisons.htm*

But Wait, There's More...

Next issue we will talk about setting rates, doing billing, some health insurance basics, creating and maintaining visibility, and time management.

The full bibliography for this article will be published in the next issue's installment. Folks who just can't wait can find it, as well as other useful resources, online at *http://www.virtual.net/Ref/resources.html.*

here comes the grooming

This is another in our series of articles about the many ways that organizations resemble other living things. As we pointed out in our last article, since people are primates, organizations are particularly likely to display primate-like behavior. Today's column involves grooming behavior. Monkeys spend a lot of time grooming one another. They do indeed pick out the occasional burr or louse from one another's fur. Those who study primates believe that grooming behavior also satisfies deeper emotional needs, bonding individuals together for mutual care, and, ultimately, mutual protection. Monkeys do it because it makes them feel good.

So it is not surprising that we find grooming behavior in organizations. It can be rather subtle, however, and even metaphorical in some ways. The yearly performance reviews that many organizations carry out are a kind of grooming behavior. The organization examines itself carefully, cleaning house as needed, and smooths its corporate fur. And collectively it feels better when the job is done.

More subtly, many individuals every day do things that enhance and support the organization. At one company in a state of rapid change, it seemed every meeting started with 20 minutes of news and gossip, as people found out who was doing which job, who had left, who had changed jobs, which projects were slipping, etc. It was

by Steve Johnson

Steve Johnson has been a technical manager on and off for nearly two decades. At AT&T, he's best known for writing Yacc, Lint, and the Portable Compiler.

scj@mathworks.com

and Dusty White

Dusty White works as a management consultant in Silicon Valley, where she acts as a trainer, coach, and troubleshooter for technical companies.

dustywhite@earthlink.net

almost impossible to start the business of the meeting until this had been gone through. When it had, however, people relaxed and felt better, and they could focus on the subject of the meeting.

Organizations also encourage grooming behavior among employees. In fact, we may speak of someone who is being "groomed for upper management." The organization has singled out this individual and is taking particular care to get the burrs and lice out of their fur. Employees groom their managers, very obviously in the case of some "brownnosers" or by more subtle use of body language, inflection, and voice in the manager's presence.

Look at a company just after a major reorganization is announced. The topic is on every lip. There is discussion of the causes, who knew about it early, who anticipated it, who is now "out" and who is now "in," who is doing which job now, and so on. The monkey just got his fur rumpled and needs some serious grooming. Over the next several days, the organization gets its new coat of fur in order and reverts to more ordinary behavior.

Sometimes, this grooming goes to remarkable lengths. One of us once attended a meeting with 150 or so people who had been flown in from 10 states and three foreign countries. The content of the meeting was well known and understood by all the employees. I personally learned not one single thing at the meeting. The big-cheese manager spoke along with all of his direct reports. The most favored of his direct reports spoke for nearly an hour. Others spoke for less time. When I realized that my boss was only allotted 12 minutes, I knew his days were numbered (he was demoted three months later).

At the time I was stunned that an otherwise sensible and profitable company would spend the better part of a quarter million dollars in salary and travel money to have an all-day meeting that conveyed no new information. And then I realized — this was grooming the organization. The big cheese had his cheeseness acknowledged by his entire organization. The smaller cheeses had their places validated in front of the whole organization. The attendees overcame their jet lag and ennui enough to be treated to several good catered meals. Grooming.

When monkeys groom one another, they aren't out gathering bananas. When companies groom themselves, they are not building product or satisfying customers. Grooming probably does make the organization stronger in the long run, if not carried to excess. In any case, a certain amount of it is probably inevitable, and worthy of the same kind of amused tolerance we give our appendix and coccyx – holdovers from our animal ancestors, of no particular utility, but easy to live with when not broken or infected.

jack-of-all-trades, master of none

Opinion

by Carl Shogren

Carl Shogren is currently the Senior UNIX System Administrator for AGCO-Corporation, one of the world's largest manufacturers, designers and distributors of agricultural equipment. As a "jack-of-all-trades" he also wears the hat of Backup Oracle DBA.

shogrence@hotmail.com

I was deep in a discussion the other day with a friend's brother-in-law, who also happens to be the day-to-day administrator for a hospital's Webmail server, when I was told something I had to disagree with. This "maintainer sysad" told me that he was frustrated with the fact that the hospital's administration expected him to "know everything" instead of letting him concentrate on the particular functions of his job. My immediate reply was that this is what had attracted me to the system administration/analyst career path in the first place.

I have always enjoyed having to learn something different with each new task I have been assigned. The initial planning, installation, and implementation of the hardware/software required to perform a task has always been my greatest joy. Sure, each new undertaking involves the risk of being a "novice," but if you have the "sysad gene," as many former colleges of mine have called it, you quickly become a "Jack of the trade," implementing a new system only to leave it later for someone to keep the system running as the "Master" of daily server administration. I have found that to set up a "Sendmail" MTA server or an "Apache" Web server is different with each installation. Different security requirements, hardware limitations, and financial considerations, which really affect all parts of a system, have to be dealt with each time I start a new task. Even the eventual training of the maintainer who will ride shotgun on the system after I leave is different each time, owing to the individuality of people and their varying capabilities.

The pitfalls of this career choice have been as many and as varied as the jobs themselves. My training a 9–5 Monday through Friday IT department employee in the basics of daily server maintenance, and making the tasks involved look easy, has lead more than one unknowledgeable job site manager to wonder why he had hired the overpriced original server creator to begin with. I cannot count how many times I have gone into a situation and asked why something was being done a particular way only to get the pat "If it works don't try to fix it" or the ever popular "That's the way it's always been done" replies.

This has made me realize that the knowledge to maintain and keep the system running was in place but not the knowledge to change, upgrade, or even conform the system to new job requirements. Examples like this have lead to a growing data consulting field worldwide, with 9–5 IT departments as the primary customer targets. Consultants are great, but with the rapid advancement of technology, both hardware and software, any large or even small company that depends on its IT department for "business," from actual customer products to customer billing, should realize the necessity of having at least one or two of these highly paid "sysad gene"-enabled, 60+ hour-a-week analysts on their permanent payroll. Sorry all you wealthy consultants! Let's admit it: the sysad gene isn't required for an IT employee to follow a well-written manual on day-to-day maintenance of an Oracle Database server that was set up so well that it hasn't been down for other than routine maintenance in two years. But try to get this same step 1, step 2 "by the manual" employee to fine tune this same server after a new firewall has been placed between the database clients and itself and too often the result is a blank stare accompanied by a willingness to blame any problems on the hardware or software.

In a field that generates the sort of salaries and opportunities that ours does, I think a distinction needs to be made between system administrators/analysts who only want to specialize in their known daily IT functions, although these are valid and required functions, and the gene-enabled sysads who are always looking forward to the next unknown challenge. Specialization is good, but locking oneself down to the particular task that your present job requires makes you more of a system operator, once again not a bad thing, than a system administrator.

To finish this thought, the conversation with the Webmail server administrator originally started because I showed him how his system had been hacked by a well-known vulnerability of his particular software/hardware implementation. A vulnerability that wasn't in his daily operations manual. 'Nuff said.

the bookworm

by Peter H. Salus

Peter H. Salus is a member of the ACM, the Early English Text Society, and the Trollope Society, and is a life member of the American Oriental Society. He is Chief Knowledge Officer at Matrix.Net. He owns neither a dog nor a cat.

peter@matrix.net

Wow! What a year!

It's really tough for me to contemplate the end of the year and holidays. I'm writing this the first week in October, less than a month after September 11. But we can hope that 2002 will be a happier year than 2001. As usual, my 10 best for the year are at the end of this column.

Penguin Care

I really like Gagne's book. But I need to confess that I was one of those Addison-Wesley had read the manuscript, and I'm thanked in the acknowledgments. I liked the first chapters when I saw them; I like the finished book.

Anyone who's read Gagne's columns in *Linux Journal* knows that he's knowledgeable, witty, and jocular. The volume reflects all these aspects of his personality. The book is well organized, yet I'm not certain that it's really suitable for a raw beginner. I think that if you are more than a real newbie, this is the very best book on Linux system administration I have seen. And it is up-to-date but eschews vendor specificity.

Really nice job, Marcel.

Phone Wires

IMP #5 was intended for the Harvard Science Center in January 1970. But the phone company had a "problem" running a dedicated connection from Harvard to BBN (both in Cambridge, Massachusetts). It was the first Net-telco problem. It was not the last.

Yet there are few books on Net telephony (not VoIP). Gast's "survival guide" is an excellent one. A T1 is more than just a wire that plugs into an alien box which then connects to a router. In view of the fact that most *;login:* readers encounter T1, T3, etc. far more than they deal with dialup, I think that Gast has supplied something worthwhile and needed.

T1 has a caribou on the cover.

Networks

Sloan's Tools is another neat book from O'Reilly. I especially liked his tool approach (after all, that's what I consider one of the most important and distinctive features of UNIX). His list of tools and sources – from Analyzer and Argus through MRTG and nemesis to xplot and xv – is simply superb.

A basilisk adorns Sloan's book.

Web Sociology

Huberman's Laws is an interesting 100-page exposition of the surprising regularities that show up in Web usage. Among the millions of Web sites and the many millions of pages, there are pathways and agglomerations and other patterns. What Huberman has produced is a fascinating analytic essay on social dynamics and group strategy. While not "technical," it's well worth reading.

And Business

The second edition of Chase and Shulock carries a "seal" on the cover, proclaiming "Essential Tips for Surviving the Dot-Com Fallout!" I'm not so sure. But I may just be the Grinch at their Christmas.

I found reading Chase and Shulock quite interesting, even though they seem to confuse the Web and the Internet it "rides" on: but it's unclear to me that they either achieve their purpose or

BOOKS REVIEWED IN THIS COLUMN

LINUX SYSTEM ADMINISTRATION
Marcel Gagne
Boston: Addison-Wesley, 2001. Pp. 532
ISBN 0-201-71934-7

T1: A SURVIVAL GUIDE
Matthew S. Gast
Sebastopol, CA: O'Reilly & Associates, 2001
Pp. 288. ISBN 0-596-00127-4

NETWORK TROUBLESHOOTING TOOLS
Joseph D. Sloan
Sebastopol, CA: O'Reilly & Associates, 2001
Pp. 346. ISBN 0-596-00186-X

THE LAWS OF THE WEB
Bernardo A. Huberman
Cambridge, MA: MIT Press, 2001. Pp. 105
ISBN 0-262-08303-5

ESSENTIAL BUSINESS TACTICS FOR THE NET
Larry Chase & Eileen Shulock
2nd ed. New York: John Wiley, 2001. Pp. 315
ISBN 0-471-40397-0

enable the increased productivity or more effective marketing they claim they do.

Top 10 for 2001 (in no particular order):

Russell C. Pavlicek, *Embracing Insanity* (SAMS)

Lincoln Stein, *Network Programming with Perl* (Addison-Wesley)

Jim Mauro & Richard McDougall, *Solaris Internals* (Prentice Hall)

Martin Dodge & Rob Kitchin, *Mapping Cyberspace* (Routledge)

Aviel D. Rubin, *White Hat Security Arsenal* (Addison-Wesley)

Paul Albitz & Cricket Liu, *DNS and BIND* 4th ed. (O'Reilly)

Charles E. Perkins, ed., *Ad Hoc Networking* (Addison-Wesley)

Marcel Gagne, *Linux System Administration* (Addison-Wesley)

Joseph D. Sloan, *Network Troubleshooting Tools* (O'Reilly)

Thomas H. Cormen et al., *Introduction to Algorithms* 2nd ed. (MIT Press)

Book Reviews

WIRELESS WEB (A MANAGER'S GUIDE)

Frank P. Coyle

Boston: Addison-Wesley, 2001. Pp. 248
ISBN 0-201-72217-8

Reviewed by Ulrich Weis
uw@saar.de

A first glance at the book satisfies my prejudices: small book (just about 250 but thick pages), nice cover, a font appearing to be a bit bigger than normal (managers tend to be older than the average Joe Hacker), broad right margin (filled with only some remarks), a lot of nice images and charts, and even some very detailed advice about who should read or skim which chapters. Definitely for managers - let's see if that counts for the content too.

Wireless Web is divided into nine chapters, a 35-page glossary, and an intense index. Each chapter starts with a short overview, continues by treating the topic from the general to the specific, gives a short summary, and finishes with printed and/or Web resources.

Chapter 1 ("The Wireless Web") introduces the reader to the overall wireless area, pointing out hype cycles, application opportunities, and technology enablers. More technical stuff (devices like PDA, cell phone, pager) is handled in Chapter 2. Chapters 3 and 4 introduce Bluetooth, a technology for connecting both IT-equipment over short distances "at chance" and wireless LAN (WLAN), which are mostly used as replacements for wired networks.

"Networks and the Quest for Bandwidth" is the title of Chapter 5, dealing with second- and third-generation wireless networks (2G/3G cell phones). Coyle not only reports the standards used in the world, but talks about politics and migration (2G -> 3G).

Chapters 6 through 8 are on protocols and languages, dealing with WAP, XML, and Java. XML (and all its subsidiaries) is especially in the fireline of big companies, as it's going to be the "standard" for "wireless content." And content is what makes those products worth livinghaving.

The last chapter is probably the most important subject covered: security. Coyle discusses requirements, threats, signatures, encryption, VPN, and all that. While the chapter is good at what it's presenting, this reviewer is feels it really misses some points. One is the biggest security problem of all: the user. It definitely doesn't help you to secure access if your boss loses his (her) PDA with passwords written on some tape on the back. And, up to now, no manufacturer has apparently thought about allowing a user to "self-destroy" data on PDAs, if there were several unauthorized access attempts.

Another point of concern, especially to Europeans, is backdoor-access or even hacking by "governmental institutions." "Echelon" is one of the keywords that come to mind, at least to security-concerned Europeans. Coyle doesn't address this point at all.

As a non-US-citizen, this reviewer must point out that most of the information is generally useful in all countries but that some stuff should definitely never be used, at least in Europe (900 MHz cordless phones, for example, disturb air traffic frequencies). There are other mistakes, but only minor ones: for example, a reference for a Bluetooth book in Chapter 4, which should be in Chapter 3.

Overall, *Wireless Web* does a good job; it is well written and easy to read. Just use it for the purpose it's been created for: if your CEO bores you again with questions about WAP, SMS, Bluetooth and all those buzzwords, hand over this book. This will probably give you some peace. At least until your boss wants to use all that nice stuff.

To sysadmins, the book offers a short but mostly complete overview of wireless technology. You'll probably find some helpful data in the large number of URL references or the tables presenting a lot of technical details (standards, comparisons).

The book is not just marketing talk but gives some real technical overview.

USENIX news

Technical Maturity, Reliability, Implicit Taxes, and Wealth Creation

by Daniel Geer

President, USENIX Board of Directors

geer@usenix.org

Stu Feldman, now head of computer science research for IBM but also once part of the original UNIX team at Bell Labs (he's author of, for example, make) used to illustrate his talks on technical maturity with what I remember as a simple five point scale:

1. you had a good idea
2. you could actually make it work
3. you could convince a friend to try it
4. people no longer asked why you were doing this
5. other people got asked why they weren't doing it.

I like that. It is easy to remember, and I have five fingers the better to count it off. It also makes sense. In the way I am using it here, many of the technologies that dominate our work lives as USENIX members are now generally important to the world at large, which is what I wrote about last month. As a USENIX partisan and in the context of this article, I'd observe that really a lot of things got their level 1 start amongst our members, showed up at a USENIX WIP at level 2 or 3, showed up again as a paper at around level 4, are now level 5 worldwide.

Sometimes it is not technologies, per se, but ideas or whole fields that move along Feldman's scale. We've played there, too; from UNIX and its mindset to client-server applications to clustered computing models to network security to mobile computing to filesystems to various programming languages to whatever. If I had to do Feldman's scale over again, I might say that a mature technology is one where its reliability is (has become) the principle metric against which its price is calibrated.

Reliability is certainly a hallmark of critical infrastructures. A lot of us are paid to deliver reliability of large systems. Complexity is our enemy yet, if truth be told, complexity is part of the reason forour employment. I happen to work in security, and it is clearly true there, even if a large part of the complexity is itself a consequence of a market demand for power and convenience.

For the most mature market sectors, operational failure is operational failure; it leads to the same bad things regardless of whether the cause of the operational failure was electric power, hacker invasion, product liability, systems administration confusion or just plain bad luck – it won't matter. For example, reliability as seen from an operational risk point of view is about to be enshrined in formal regulation for the banking sector. The so-called Basel Capital Accord, ordinarily too esoteric a matter to discuss in the USENIX context, is under a revision process that will eventually lead to banks being required to set aside capital not only for credit risk and market risk but also operational risk. Capital set aside is a way of preparing for unexpected losses that might otherwise challenge the safety of the bank. Capital set aside is capital that is not earning money. Capital set aside to cover unexpected losses due to operational failure is a tax on the wealth-creating power of the bank. This is the kind of thing that gets attention at the Board of Directors level. This puts

operational risk directly in the line of fire when the Board bears down on senior management's ability to create shareholder value.

Well, this is not just about banks. In a late-October *NY Times* editorial, Richard Berner (chief US economist for Morgan Stanley) wrote:

> But in the long term, terrorism is imposing new costs that are unlikely to go away. For every business, insurance and security costs will be higher. For many, the benefits of just-in-time management will be sacrificed as companies hold more inventory to guard against breaks in the global supply chain. The threat of cyberterrorism, which once seemed distant, will almost certainly lead to new measures for Internet security, slowing activity even for those operating in the supposedly frictionless world of cyberspace. And America's first experience with bioterrorism has thrown sand in the gears of commerce, government and everyday life, requiring new caution and precaution in once-mundane activities like mail sorting.
>
> Together, those costs could represent a new form of supply shock, like a longer-term tax on the economy that will hurt growth and could boost inflation. That would be a toxic combination for global financial markets.

The reason I quote this at length is simple – the very kind of stuff that we all do, whether it be nameservice maintenance, application or network security, systems administration with complex authorization and systems segregation constraints, hardware and software deployment under a skeptical eye, UI design intended to transmit power but not risk, user account administration, or whatever – it is about to be at once faced with very, very much higher standards for what constitutes probity and best practices while at the same time it is, and will be seen to be, a tax on productivity growth, i.e., what we do will be an enemy of wealth creation. Our new marching orders are tough, for the principle requirement will be that of reliability and the costs for this kind of reliability will not be trivially fixed by cleverer programming.

All of us have to look at this and look at this hard. The world profited as a whole from structural changes in the rate of productivity growth in the 90s that come directly from information technology investment on a grand scale. If we are to again achieve growth rates that are wealth creating on a broad scale, we have got to deliver reliability in a way that enhances productivity growth, not at the expense of it. This is the very eye of the storm.

2002 Election for Board of Directors

Ellie Young
Executive Director

The biennial election for officers and directors of the Association will be held in the Spring of 2002. A report from the Nominating Committee will be posted to *comp.org.usenix* and the USENIX Web site in mid-December, and also published in the February issue of *;login:*.

Nominations from the membership are open until January 11, 2002. To nominate an individual, send a written statement of nomination signed by at least five (5) members in good standing (or five separate nominations), to the Executive Director at the Association office, to be received by noon, PST, January 11, 2002. Please include a Candidate's Statement and photograph to be included in the ballots.

Ballots will be sent to all paid-up members on or about February 8. Members will have until March 22 to cast their vote. The results of the election will be announced in *comp.org.usenix*, the USENIX Web site, and in the June issue of *;login:*.

The Board is made up of eight directors, four of whom are "at large." The others are the President, Vice President, Secretary, and Treasurer. The balloting is preferential; those candidates with the largest number of votes are elected. Ties in elections for Directors shall result in run-off elections, the results of which shall be determined by a majority of the votes cast. Newly elected directors will take office at the conclusion of the first regularly scheduled meeting following the election, or on July 1st, whichever comes earlier.

International Olympiad of Informatics 2001

by Don Piele
USACO Director
piele@cs.uwp.edu

With all four members earning medals, the US team recently completed its most successful showing ever at the International Olympiad in Informatics. The competition, featuring teams from 74 countries, was held in Tampere, Finland, July 14 to 21.

Reid Barton, a home schooled high school senior from Arlington, Mass., was the top overall contestant of the entire event. Barton,s total score was 55 points better than the next highest competitor, the largest margin of victory in Olympiad history, earning him his second gold medal in as many years. The week prior to the Computer Olympiad,

Barton had won his fourth consecutive gold medal at an international math competition.

Two US team members, Tom Widland of Albuquerque, NM, and Vladimir Novakovski of Springfield, VA, earned silver medals in the competition. Steven Sivek of Burke, Va., captured a bronze medal. Widland is a senior at Albuquerque Academy; Novakovski and Sivek both are juniors at Thomas Jefferson High School for Science and Technology in Springfield, Va.

The US team was sponsored by USENIX.

My trip report for IOI 2001 is available directly at: *http://www.uwp.edu/academic/mathematics/usaco/2001/ioi/report.htm*

All 205 photos that I took at IOI 2001 are stored can be viewed at: *http://www.ofoto.com/BrowsePhotos.jsp?collid=73793377103*

Thank you USENIX for sponsoring the USACO.

VECPAR 2002 – Announcement and Call for Papers

5th International Meeting on High Performance Computing for Computational Science June 26–28, 2002

Faculdade de Engenharia da Universidade do Porto, Porto, Portugal

http://www.fe.up.pt/vecpar2002

Important Information

Deadline for submissions: December 14, 2001

Proposals for tutorials due: December 14, 2001

Author's notification: March 8, 2002
Tutorials: June 25, 2002

Secretariat: *congress.porto@abreu.pt*

Organisation: *vecpar2002@fe.up.pt*

Topics of Interest

- Cluster and Grid Computing
- Computing in Biosciences
- Concurrent Engineering
- Data Processing
- Educational Issues in Computational Science and Engineering
- Large Scale Simulations in all areas of Engineering and Science (e.g., Computational Fluid Dynamics, Crash and Structural Analysis, etc.)
- Numerical Methods (PDE, Linear and Non-Linear Algebra, etc.)
- Parallel and Distributed Computing
- Problem Solving Environments
- Scientific Visualization

Invited Speakers

- Yutaka Akiyama (Computational Biology Research Center, Japan)
  "Human Genoma"
- Leif A. Eriksson (Uppsala University, Sweden)
  "Computational Chemistry"
- Vipin Kumar (University of Minnesota, USA)
  "Data Mining"
- Rainald Lohner (George Mason University, USA)
  "Computational Fluid Dynamics"
- Ed Seidel (Max-Planck-Institut für Gravitationsphysik, Germany)
  "Problem Solving Environment"

Proceedings

Proceedings, including full text of all presentations, will be available during the meeting. Additionally, a book will be published by Springer in its Lecture Notes in Computer Science series (*http://www.springer.de/comp/lncs/index.html*) and distributed after the conference. This book will include the invited talks and a set of selected papers.

Years and Years

by Peter H. Salus
USENIX Historian
peter@matrix.net

On September 8 or 9 (depending on where you were on the globe) the UNIX clock ticked its 10^9 second. I was in Copenhagen at uptime(1), a terrific bash sponsored by the DKUUG. Talks (including mine and Rob Pike's), sit-down dinner, a rock group, a techno group, a multimedia show, champagne, and fireworks. Simply wonderful.

I got home in the wee hours of September 11 and got to my office, bleary-eyed, just after the attacks. Not the best greeting.

But, two weeks later, I was in Sydney as keynote for the AUUG's meeting (the other two keynoters were Evi Nemeth and Rob Kolstad). It was 25 years since John Lions installed v6 on the PDP-11 at UNSW. Another wow.

In November I'll be at the ALS; it's 10 years since Linus posted the 0.01 kernel on Helsinki's FTP site. In December I'll be at LISA talking about all the anniversaries that took place in 2001: 125 years of the telephone, 50 years since the first commercial computer went on sale (about the size of a small garage, 5,000 tubes, water-cooled). UNIVAC sold 46 of them at $1 million each. Think of it as you heft your laptop or PDA.

And the ARPANET, which had four machines at the end of December 1969, now has about 160 million.

Happy New Year!

Announcement and Call for Papers **USENIX**

11th USENIX Security Symposium

http://www.usenix.org/events/sec02

August 5-9, 2002 **San Francisco, California**

Important Dates for Refereed Papers

Paper submissions due: *January 28, 2002*
Author notification: *March 25, 2002*
Camera-ready final papers due: *May 13, 2002*

Symposium Organizers

Program Chair
Dan Boneh, *Stanford University*
Program Committee
Steve Bellovin, *AT&T Labs–Research*
Matt Blaze, *AT&T Labs–Research*
Drew Dean, *SRI International*
Kevin Fu, *M.I.T.*
Brian LaMacchia, *Microsoft Corporation*
Patrick Lincoln, *SRI International*
Vern Paxson, *ACIRI/ICSI*
Radia Perlman, *Sun Microsystems Laboratories*
Mike Reiter, *Bell Labs, Lucent*
Avi Rubin, *AT&T Labs–Research*
Adam Stubblefield, *Rice University*
Leendert van Doorn, *IBM T.J. Watson Research Center*
Wietse Venema, *IBM T.J. Watson Research Center*
Dan Wallach, *Rice University*
Bennet Yee, *University of California, San Diego*
Elizabeth Zwicky, *Counterpane Internet Security*
Invited Talks Coordinator
Dan Wallach, *Rice University*

Symposium Overview

The USENIX Security Symposium brings together researchers, practitioners, system administrators, system programmers, and others interested in the latest advances in security of computer systems.

If you are working on any practical aspects of security or applications of cryptography, the program committee would like to encourage you to submit a paper. Submissions are due on January 28th, 2002.

This symposium will last for four and a half days. Two days of tutorials will be followed by two and a half days of technical sessions including refereed papers, invited talks, Work-in-Progress reports, Birds-of-a-Feather sessions, and panel discussions.

Symposium Topics

Refereed paper submissions are being solicited in all areas relating to systems and network security, including but not limited to:

- Adaptive security and system management
- Analysis of malicious code
- Analysis of network and security protocols
- Applications of cryptographic techniques
- Attacks against networks and machines
- Authentication and authorization of users, systems, and applications
- Automated tools for source code analysis
- Denial-of-service attacks
- File and filesystem security
- Firewall technologies
- Intrusion detection
- Privacy preserving systems
- Public key infrastructure
- Rights management and copyright protection
- Security in heterogeneous environments
- Security of agents and mobile code
- Security of Internet voting systems
- Techniques for developing secure systems
- World Wide Web security

Since USENIX Security is primarily a systems security conference, papers focusing on cryptographic primitives or electronic commerce models are encouraged to seek alternative conferences.

Refereed Papers (August 7-9)

Tutorials, Invited Talks, WiPs, and BoFs

Tutorials (August 5-6)
Invited Talks (August 7-9)
Panel Discussions (August 7-9)
Work-in-Progress Reports (WiPs)
Birds-of-a-Feather Sessions (BoFs)

For full information consult the Web at
http://www.usenix.org/events/sec02/

NordU2002 - The fourth NordU/USENIX Conference
February 18–22, 2002, Helsinki, Finland

February 18-20

TUTORIALS

14 tutorials such as:

- To BGP or Not to BGP:
  Making the internet Connection
  Instructor: Vincent C Jones,
  Networking Unlimited
- Sendmail Configuration and Operation
  (Updated for Sendmail 8.12)
  Instructor: Eric Allman, Sendmail
- UNIX Kernel Internals:
  Data Structures and Algorithms
  Instructor: Marshall Kirk McKusick
- Inside the Linux Kernel
  Instructor: Theodore Ts'o
- Advanced Topics in DNS Administration
  Instructor: Jim Reid, Nominum

February 21-22

CONFERENCE and EXHIBITION

25 presentations with topics as:

- Security
- FreeNIX
- Refeered Papers
- WWW and scripting
- Thin Clients
- Storage and Clustering
- Misc
- Sponsors presentations

Some of the speakers are:

Solar Design, Paul-Henning Kamp, Ken Coar, Serge Robe, David Boyes, Guido van Rooij, Paul Massiglia, Jonathan Appavoo, Mattias Ettrich, Werner Koch etc.

KEYNOTE SPEAKERS

Gary McGraw, Cigital's Chief Technology Officer

Building Secure Software
How to Avoid Security
Problems the Right Way

Peter H. Salus, Chief Knowledge Officer of Matrix.Net

Unix and its Children

Bruce Perens, Hewlett Packard.
Primary author of the Open Source Definition

Open Source, Standards, and Networks:
Tools of Liberty and Democracy

Gold sponsor

hp invent

Silver sponsors

HITACHI DATA SYSTEMS

VERITAS

Exhibitors

Aurora Software Inc, Helsingin DataClub Oy, Hewlett Packard Oy, Hitachi Datasystems AB, Raditex AB, Veritas Software AB

http://www.nordu.org/NordU2002/

.SE EurOpen

DKUUG EurOpen

NUUG

FUUG

CONNECT WITH USENIX & SAGE

MEMBERSHIP, PUBLICATIONS, AND CONFERENCES

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: +1 510 528 8649
FAX: +1 510 548 5738
Email: <office@usenix.org>
<login@usenix.org>
<conference@usenix.org>

WEB SITES

<http://www.usenix.org>

<http://www.sage.org>

EMAIL

<login@usenix.org>

COMMENTS? SUGGESTIONS?

Send email to <ah@usenix.org>

CONTRIBUTIONS SOLICITED

You are encouraged to contribute articles, book reviews, photographs, cartoons, and announcements to *;login:*. Send them via email to *<login@usenix.org>* or through the postal system to the Association office.

The Association reserves the right to edit submitted material. Any reproduction of this magazine in part or in its entirety requires the permission of the Association and the author(s).

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

;login:

PERIODICALS POSTAGE
PAID
AT BERKELEY, CALIFORNIA
AND ADDITIONAL OFFICES

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

POSTMASTER
Send address changes to *;login:*
2560 Ninth Street, Suite 215
Berkeley, CA 94710