;login:

THE MAGAZINE OF USENIX & SAGE

February 2003 • volume 28 • number 1

GALA SYSTEM ADMINISTRATION ISSUE

inside:

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

USENIX Upcoming Events

4TH USENIX SYMPOSIUM ON INTERNET TECHNOLOGIES AND SYSTEMS (USITS '03)

MARCH 26–28, 2003
SEATTLE, WASHINGTON, USA
http://www.usenix.org/events/usits03

2ND USENIX CONFERENCE ON FILE AND STORAGE TECHNOLOGIES (FAST '03)

MARCH 31–APRIL 2, 2003
SAN FRANCISCO, CALIFORNIA, USA
http://www.usenix.org/events/fast03

THE FIRST INTERNATIONAL CONFERENCE ON MOBILE SYSTEMS, APPLICATIONS, AND SERVICES (MOBISYS '03)

Jointly sponsored by ACM SIGMOBILE and USENIX in cooperation with ACM SIGOPS

MAY 5–8, 2003
SAN FRANCISCO, CALIFORNIA, USA
http://www.usenix.org/events/mobisys03/
Camera-ready final papers due: March 4, 2003

THE 9TH WORKSHOP ON HOT TOPICS IN OPERATING SYSTMES (HOTOS IX)

Sponsored by USENIX in cooperation with IEEE TCOS

MAY 18–21, 2003
LIHUE, KAUAI, HAWAII, USA
http://www.usenix.org/events/hotos03/
Notification of acceptance: March 17, 2003

2003 USENIX ANNUAL TECHNICAL CONFERENCE

JUNE 9–14, 2003
SAN ANTONIO, TEXAS, USA
http://www.usenix.org/events/usenix03/
Camera-ready final papers due: April 8, 2003

ACM/IFIP/USENIX INTERNATIONAL MIDDLEWARE CONFERENCE

Sponsored by ACM, IFIP, and USENIX

JUNE 16-20, 2003
RIO DE JANEIRO, BRAZIL
http://middleware2003.inf.puc-rio.br/

2003 LINUX KERNEL DEVELOPERS SUMMIT

JULY 11-22, 2003
OTTAWA, CANADA

12TH USENIX SECURITY SYMPOSIUM

AUGUST 4–8, 2003
WASHINGTON, DC, USA
http://www.usenix.org/events/sec03/
Notification of Acceptance: March 20, 2003
Camera-ready final papers due: May 13, 2003

BSDCON '03

SEPTEMBER 8–12, 2003
SAN MATEO, CALIFORNIA, USA
http://www.usenix.org/events/bsdcon03/
Paper abstracts due: April 1, 2003
Notification of Acceptance: May 12, 2003
Camera-ready final papers due: July 8, 2003

17TH SYSTEMS ADMINISTRATION CONFERENCE (LISA '03)

Sponsored by USENIX, The Advanced Computing Systems Association and SAGE, The System Administrators Guild

OCTOBER 26-31, 2003
SAN DIEGO, CA, USA
http://www.usenix.org/events/lisa03
Paper submissions due: April 21, 2003

For a complete list of all USENIX & USENIX co-sponsored events, see
http://www.usenix.org/events

contents

;login: Vol. 28 #1, February 2003

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

;login: (ISSN 1044-6397) is published bi-monthly 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.

Cover: LISA XVI Program Chair Alva Couch
See p. 73 for a report on the Conference.

motd

by Rob Kolstad

Rob Kolstad is currently Executive Director of SAGE, the System Administrators Guild. Rob has edited *;login:* for over ten years.

kolstad@sage.org

Ugly Babies

I recently attended Charlie Bass's WebDevCon in Las Vegas (held concurrently with ApacheCon). The vast majority of WebDevCon's attendees were developers of commercial software, and they pretty much used Microsoft operating systems and products exclusively. It was very enlightening for me.

As casual discussion topics rotated during one particular lunch, the others at the table turned their attention to the .NET initiative. They discussed the various tools available and offered various critiques. Some of the newer Integrated Development Environment (IDE) received high marks. One attendee made a telling remark (this is not an exact quote but is the general idea): "We have a job to do. We all know multiple programming languages and can learn new ones fairly rapidly. We choose a good tool and work with it. I am particularly pleased at my new ability to create prototypes very rapidly."

Rapid prototyping. He wasn't talking in the context of "Extreme Programming" (see *http://www.jera.com/techinfo/xpfaq.html* for an excellent brief introduction to "XP"). He was extolling the same virtues that the Perl, Python, and other aficionados enjoy when they suggest bringing up quick-and-dirty versions of software to see if it solves the problem-at-hand.

The discussion then turned to the pivotal comment: "Yeah, it sure saves us from ugly baby syndrome." My mind churned trying to figure out the reference.

"No one has ever had an ugly baby," it was explained. "Maybe you or I might think the baby is less than attractive, but when it's *your* baby, it's the most beautiful baby in the world."

And the light came on for me. When programmers, committees, designers, architects, or anyone else spends a fairly long time on a project, then the results of that project are guaranteed a positive evaluation because: There Are No Ugly Babies (TANUB).

Now that I understand this, I understand my emotions when people talk to me about my programming contest grading system (aka "Rob, Junior," my baby). It's a beautiful system. Oh, maybe there's a rough edge here or there, but otherwise it's perfect. TANUB.

Or so I thought. Russ Cox, now of MIT, began an "improvement regimen" for the grading system that included such items as constructing a new (to me) sort of "jail." This "jail" knows how to check every system call (and the arguments thereto) that goes by as a contestant's program is executed and can be configured to allow or to deny the success of the call based on a number of interesting attributes of such calls. Well, that would improve my baby. In fact, Russ has a huge number of good ideas that will be improving my baby. So many that maybe, just maybe, it will become his baby. Only then will I be able to step back and see if it's ugly or not.

So what's the prescription? Well first, let's acknowledge that it's a variant of an older prescription that you've probably heard repeatedly (I think Mike O'Dell inculcated me with it): "Generally, it's OK to fail, as long as you fail quickly." This prescription's corollary is: "Never fail slowly." I think these are great words to live by. They enable one to experiment freely, as long as the experiments are not "costly" in terms of cash, prestige, data loss, etc.

The new prescription: "Don't spend so much time on the first version(s) of a project that you will be unable to judge its success or failure unemotionally." This suggests rapid prototyping as a great way to implement the mature engineer's credo: "Plan to throw the first one away." Why not? If the first one was "cheap," it won't hurt at all to discard it, redesign it, or improve it in some other way. The opposite approach is mind-numbing in its political complexity, trust me.

Best wishes to everyone for a productive New Year and many successful projects.

letters to the editor

To Robert Haskins:

I found a mistake in your article on stopping spam [*;login:* Vol. 27, No. 4, August 2002, page 48] that can produce serious adverse affects. You state:

"A connection rate throttle will limit the number of connections per second from a given server."

This is incorrect. It limits the *total* number of connections per second. So one connection from each of three different servers in a second will hit the limit. This means your advice of setting it to three will limit the machine to accepting a total of three connections per second. That will be a performance killer. Don't do that!

Gregory Neil Shapiro

To the Editor:

I am writing to you regarding the description of my article "Active Network Defense: Some Concepts and Techniques" in the "in this issue" section of the December 2002 (Security themed) issue of *;login:*. Unfortunately, the introduction suggests that my article "advocate[s] attacking the immediate surroundings of the attacker." This was not my intent at all, and I did not advocate that. I was merely "exploring what the possibilities are today."

Regards,

Sven Dietrich
spock@cert.org

Rik Farrow responds:

I agree that I misconstrued Sven Dietrich's point, that he was exploring possibilities, not actively advocating them. I got carried away. The US government, in the second draft of the document from the administration, named the National Strategy to Secure , actually goes much further than Dietrich does:

The new draft cautions that it can be difficult or even impossible to trace an attack's source. But it warns that the government's response "need not be limited to criminal prosecution. The United States reserves the right to respond in an appropriate manner, including through cyber warfare," it said.

This comment appeared in an AP Wire story written by Ted Bridis on January 6, 2003 (*http://news.yahoo.com/news?tmpl=story2&cid=528&ncid=528&e=3&u=/ap/20030106/ap_on_go_pr_wh/securing_cyberspace*). Hopefully, the US response will be limited to "cyber warfare" attacks against any hapless ISP or university involved in an alleged attack.

rik@spirit.com

Appropriate Responsibilities

by Tina Darmohray

Tina Darmohray, contributing editor of *;login:*, is a computer security and networking consultant. She was a founding member of SAGE. She is currently a Director of USENIX.

<tmd@usenix.org>

You see just about everything in the consulting business. Sometimes you see examples of "doing everything right," and those are invigorating. You leave those sites thinking, "Wow! Now that's the way a bunch of machines should be run." You revisit those sites in your mind; they are so technically correct and clean that it's a pleasure to think about, like looking under the hood of a customized hot rod and proclaiming that the engine compartment is a thing of beauty.

For every lean and mean site you have the pleasure of seeing you also come across those at the other end of the spectrum. The system administrators are disgruntled, the managers are frustrated, the machines and network are laid out in a haphazard way, and the applications are insecure and underperforming. You revisit those sites in your mind too, but to work out what went wrong, rather than to enjoy the image of a technical thing of beauty.

While perhaps unpleasant, revisiting the "wrong" sites in order to determine what brought them to where they are can be tremendously rewarding in a different way. Figuring out what got them to where they are can be instructional and prevent having to learn the lesson firsthand the hard way.

I spent quite a bit of time trying to determine what had gone wrong at one particular site. There was no apparent plan for the way things were laid out, and the symptoms were insecure applications, inefficient use of resources, and a downtrodden staff. Further inspection revealed the root of the problem: professional responsibility was left to the folks in the trenches and they failed to take it on. That's right, the system administrators in the trenches were mostly at fault for this particular SNAFU.

Wow! That's kind of harsh. A public proponent of system administrators placing the blame for a SNAFU on the good guys? When did she join the opposition? Well, I was surfing on the sageweb site the other day and I came across the SAGE Job Descriptions for an Intermediate/Advanced administrator. Under "Required Skills" it says, among other things:

- Strong interpersonal and communication skills; *capable of writing purchase justifications*, training users in complex topics, making presentations to an internal audience, and interacting positively with upper management.
- Independent problem-solving, *self-direction*.

and under "Appropriate *Responsibilities*":

- *Initiates* some new responsibilities and helps to *plan for the future of the site/network.*
- Evaluates and/or *recommends purchases*; *has strong influence* on purchasing process.

These qualities clearly describe a senior professional who is expected to be proactive, persuasive, and has a stake in the systems s/he is managing. That is, we're not talking about a naive novice who doesn't know enough to "know better." We're talking about a technically savvy individual who is capable of, and should be held responsible for, knowing the requirements and planning for the needs of the organization they're supporting. And that's where the system administrators of this particular site had failed.

Several years earlier, as the organization moved off mainframes, they had mimicked their earlier environment by purchasing large UNIX servers and "selling" space on them. As departments and services required computing resources, they came to the central IT group. The group gave out resources on an as-needed basis, filling up the servers which had been purchased based on space and load. While this approach may streamline the purchasing process for servers (just buy another one of what we already have), it fails on almost every other front.

The machines, while mostly identical from a hardware standpoint, are unique in every other way. There are Web servers, applications, databases, and core services on these systems, but they are not distributed in any predictable way. That is, Web servers share database servers, core services are colocated with applications, and any other combination that is possible. As they say on Saturday Night Live, these machines were "a floor-wax and a dessert topping." As a result, the system administrators were unable to efficiently scale their operations because they could not take advantage of the cookie cutter approach: with every machine a unique mixture of the endless possibilities, there was no way to create standard builds, for instance. When it came time to secure the most sensitive databases, this hodgepodge of systems tripped them up: with multi-tiered applications deployed on single systems, isolating the databases behind firewalls and restricting administrative access to core systems was impossible. When it comes to disaster preparedness, this approach fails again: some pivotal machines are so complex that when they fail it will almost certainly take days (and nights) of intense system administration heroics to put Humpty Dumpty back together again.

The system administrators at this site are complaining bitterly that they're understaffed. Their systems are so complicated that they're indeed difficult to manage, but I don't think I could justify additional staff in this case. Rather, these SAGE Level III system administrators should start working up to their capacity and taking responsibility for planning the future computing requirements of their organization. Instead of complacently buying another server and blindly installing the next seven requests for resources on it, the system administrators should take the initiative to understand the needs of the departments and services and size any new servers to the applications. If this includes selling the upper management on these ideas, that's in their job description too. Retroactively, they need to create a migration plan which co-locates similar applications and begins to leverage their time with standard builds and cold-swap spares for disaster readiness. Finally, once they've located Web servers with Web servers, but separately from database servers, they can secure their site from external mischief and place sensitive data behind firewalls and restrict access to such systems or applications to "need to know/administer." In short, they need to understand the requirements and specifically and proactively plan for them.

In any job we do, we all take direction from someone, but the more senior we become, the more self-directing and proactive we're required to be. When you get to that level, you can no longer expect your manager to spell out implicit tasks. Planning for resiliency, scalability, security, and efficiency are givens that are part of doing a "good job" as a more senior system administrator. In fact, they're not only "Appropriate Responsibilities," they're required.

common problems with outsourcing deals and how to avoid them

by John Nicholson

John Nicholson is an attorney in the Technology Group of the firm of Shaw Pittman in Washington, D.C. He focuses on technology outsourcing, application development and system implementation, and other technology issues.

John.Nicholson@ShawPittman.com

Outsourcing is frequently considered a panacea: A vendor promises that because of their experience, knowledge base, proprietary tools and systems, leverage with suppliers, and so on, they can do what you do better and for less money. Unfortunately, it rarely seems to work out as well as it should. Why?

Any function can be a candidate for outsourcing, provided it is not critical to what makes your company competitive in its environment. IT is generally the poster child for outsourcing because it started that way – companies outsourced their data processing to large time-share mainframes. These days, many other aspects of IT are also subject to significant economies of scale, both in terms of operation and purchasing. Also, IT is something of a black box to the rest of the organization; therefore, if the organization doesn't understand how IT works or why it's strategically important, it must be a commodity that someone else could also do. But outsourcing is not limited to IT. HR can be outsourced, as can something like claims administration for an insurance company. Even product development can be outsourced to one of the many engineering design firms that perform that function. There are marketing firms/advertising agencies that are basically outsourced marketing departments for the companies they represent. There are even some companies that use an outside law firm as their outsourced general counsel's office. It all depends on what you do, what you need, and what you're good at.

The purpose of this article is to look at some of the common, fundamental problems in outsourcing relationships, regardless of the industry or function being outsourced, and to suggest ways of structuring the relationship to increase the likelihood that it will be successful.

Problems with Outsourcing

Problem 1: Salespeople and Executives

Salespeople are good at selling. They tell executives what the executives want to hear in order for the executives to want to buy the product that is being sold. Executives like to do big, sexy projects. An outsourcing is a dramatic way to look like you are doing *something*. The executive is bringing in an Expert who is "Best in Class" to provide a service better and cheaper than the executive's company can do for itself. Who could say no?

The first big problem in outsourcing comes from the fact that the vendor's salespeople don't necessarily understand your problems – they just know that given enough time and money, their people are likely to be able to do what your company needs well enough that you won't throw them out. The executive, always under pressure to do more with less, listens to the salesperson's promises, but probably doesn't necessarily understand the intricacies of the functions that the vendor is proposing to take over. What the executive hears is that the salesperson says that the vendor can save the exec-

utive's company some large amount of money – savings that the executive can either use for other projects or simply claim credit for. The executive trusts the relationship that has developed with the salesperson, and is frequently razzle-dazzled by the high-powered support that the salesperson can bring in to boost his or her case. Who wouldn't be impressed when the CEO of the vendor calls you up to tell you how important your business would be to them?

Unfortunately, the vagueness and ambiguity of the conversations between the salesperson and the customer executive lead directly to problems 2 and 3. Furthermore, the savings that were "promised" rarely actually appear, and almost never on the scale that was promised.

Problem 2: Philosophical Differences

The second big problem with outsourcing deals, and the reason why so many of them fail (or are at least unsatisfying), is lack of communication and lack of work up-front to design the relationship. Customers and vendors approach outsourcing with two radically different philosophies, but they rarely discuss those philosophies and the impact that they will have on the relationship.

The customer expects the outsourcer to act exactly like the customer's employees act. So if the customer asks the outsourcer to do some extra work, the customer expects that the outsourcer will just prioritize the work or stay a little late and get it done at no additional cost.

The outsourcer, on the other hand, thinks that it is charging a defined price for a defined scope of work, and if the customer asks the outsourcer to do something more, it's only fair that the outsourcer get to charge for it.

The big problem is, no one really discusses this philosophical difference openly, and it festers as the outsourcer issues change order after change order asking for more money, and the customer thinks that the outsourcer is just gouging the customer because, after all, the customer's old employees wouldn't have asked for more money. Bit by bit, the relationship deteriorates.

An additional aspect to this problem is that while the company's employees would frequently go above and beyond the call of duty to make sure that problems are resolved and customer impact is minimized, the outsourcing vendor doesn't necessarily have the incentive to do that, and so perceived customer service degrades.

For example, one company I worked with recently had a few help desk personnel who would perform a sort of level 1.5 triage. If a problem couldn't easily be resolved via the phone from the help desk, one of these people would go work the problem from the deskside. These people were integral to the customer service perceived by the end users. Part of the outsourcing vendor's proposed savings was to move the help desk off-site and eliminate those deskside visits. The customer management wanted the savings and was willing to accept the loss of that level of assistance. However, from an end-user perspective, customer service was degraded. Without a sufficiently detailed communications plan that informed users that this service would be going away, the end users would perceive this as a failure of the vendor to provide the same level of customer service as they received before the outsourcing.

In such a scenario, the customer begins hearing from its users that the service provided by the vendor isn't as good as the service they received before the outsourcing, while

Who wouldn't be impressed when the CEO of the vendor calls you up to tell you how important your business would be to them?

Different philosophies + poor internal management communication + a poorly defined scope = an unhappy relationship.

the vendor is asking for additional money. The vendor, from its perspective, is providing its services in exactly the way that it told the customer it would, and is only asking for more money to cover the additional scope that the customer is requesting.

Problem 3: Ambiguous Scope

The deterioration in the customer-vendor relationship is facilitated by the third big problem in outsourcing – lack of understanding of the scope. When an outsourcing vendor submits a proposal, it describes in general terms what it is going to do, but only very, very rarely supplies a detailed description of how things will work, who has responsibility for which functions, etc. Developing such a document takes a great deal of time and effort, and requires the cooperation of the people who are either about to be outsourced or who are already overburdened running the operation. These people do not want to spend days (weeks!) in conference rooms defining the scope. They just want the vendor to get on with it. At the same time, the senior executives who have decided that outsourcing is a "Good Thing" want the deal to happen fast, generally because they have already factored the savings from the outsourcing into their next quarter's budget.

So the scope ends up being poorly defined, and each side has a different idea of what they are responsible for doing. The customer assumes that the vendor will do everything that the people who held the positions that were outsourced did – even if that stuff wasn't in the job descriptions for those functions. The vendor, on the other hand, thinks that it has specifically defined the functions that it will perform (never mind that the proposal from the vendor is generally about as specific as Swiss cheese – the vendor *thinks* it's done a thorough job) and that it has priced the services it thinks it's going to provide.

Different philosophies + poor internal management communication + a poorly defined scope = an unhappy relationship.

How to Decrease the Chance of Being Unhappy

Where a function is going to be outsourced, it is the customer's job to make sure that the contract is properly created, in order to increase the likelihood that the relationship will be a happy one.

Competitive Procurement

First, use a legitimate competitive procurement process. You will get a better deal and have negotiating leverage in a competitive procurement that you do not have in a sole source deal. This seems intuitively obvious, but I am astounded at the number of times I hear supposedly rational, educated executives say things like, "We're not going to compete the deal because if we don't make things difficult for the vendor now, they will remember that and go the extra mile for us later on." By that logic, you should pay full price (or more) for a car in the hopes of getting better service down the road.

A second part of this is that you have to legitimately want the best deal to win. If you have already decided which vendor you want, but you are just using the other vendor(s) to drive down your selected vendor's price, all of the vendors will know and it will not work. The less preferred vendors will not put in the effort or resources because they know that they do not have a fair chance of getting the deal.

To establish a good competitive procurement, you will need to understand the scope of what you want to outsource and document your requirements as part of an RFP. The

RFP should clearly describe what you want the vendor to do, detail any assumptions that you want the vendor to make, and provide a clear timeline. The RFP should be structured so that it will be relatively easy for you to compare the proposals from each vendor. The pricing provided by the vendors, in particular, should be broken down to a level that makes "apples to apples" comparisons possible.

At the time you begin your RFP process, you should also look at the service levels that you will want for the outsourced services, and if you aren't already tracking them, you should start. Most vendors want to see at least six months' worth of performance data before they will agree to a service level, and if you start recording at the beginning of the RFP process, you are likely to have enough data to make the vendors comfortable.

Clearly Document Scope, Service Levels, and Pricing

Second, document the scope, service levels, and pricing up-front. The scope should be sufficiently detailed that it is absolutely clear who is responsible for doing what. While it does not need to reach the level of desk procedures, the scope document should be fairly thorough.

The scope should clearly define what the vendor is responsible for doing and when, and the service levels should be tailored to measure the vendor's performance of those responsibilities. If a service level measures an activity that is performed jointly or that requires some activity by the customer before it can be completed, then the situation is ripe for finger-pointing between the vendor and customer if the service level is not met.

There should be no "assumptions" in the final agreement. "Assumptions" are vendor code for "If this doesn't turn out to be true, the price will change." Any assumption should be discussed in detail, and the impact to the price or services of variations in that assumption should be clearly documented. For example, an assumption that the volume of moves will be 10,000 per year simply says that if it's not 10,000, something will change. The vendor will interpret this as, "I will be paid for a minimum volume of 10,000 moves, regardless of whether or not they happen, and I will get an additional charge for each move over 10,000." The customer position, on the other hand, is more likely to be, "If I require fewer than 10,000 moves, the vendor isn't having to do the work, so I shouldn't be charged for it, and if I do a few more, the vendor should be able to absorb those in the capacity that is already in place." The final agreement should *clearly* specify what happens to the pricing if the required number of moves is different from 10,000. The scope, service levels, and pricing should all be completed *before* the contract is signed, which leads us to . . .

Due Diligence

Third, all vendor due diligence should be completed prior to contract signing. The vendor should not be able to "re-open" the deal after the contract has been signed. The only exception to this might be if you don't have sufficient data regarding service levels. In that case, you and the vendor can agree to interim service levels while the vendor monitors its actual performance over a specified period to set the permanent service levels.

Deal Consultants

The previous three suggestions are all things you can do on your own, but if you use an experienced deal consultant and get them involved before you send out an RFP, these steps will all be much easier. For one thing, the deal consultant will have more

"Assumptions" are vendor code for "If this doesn't turn out to be true, the price will change."

The majority of problems with outsourcing deals are caused by poor communication and lack of effort early in the process.

experience evaluating and comparing the proposals from the vendors. Also, the deal consultant will understand the industry, in terms of what are reasonable positions and what are not, and also understand the pitfalls of structuring these types of deals. Vendors have people who do nothing but negotiate these deals, and if you try to go up against them without an experienced deal consultant with you, you are working under a handicap for two reasons: (1) the vendor negotiator is very experienced and practiced at negotiating deals that maximize the benefit to the vendor while minimizing its risk and liability, while you are at a disadvantage because of your relative inexperience; (2) the vendor has a team of people who can spend almost full time working on the project, while you and your people also have to keep your business running. A deal consultant can be the additional resource that helps balance out the vendor's resource advantage.

Conclusion

The majority of problems with outsourcing deals are caused by poor communication and lack of effort early in the process. As with any relationship, communication and understanding of mutual expectations are key to the ongoing health of the relationship. Customer executives considering an outsourcing need to understand what they are trying to achieve and be willing to put the effort in at the outset to increase the likelihood of getting what they want.

Negotiating outsourcing deals is not easy. You are designing a relationship that will last for five or more years, and you are attempting to build in protections for both sides for all of the things that might change during that time. Documenting the full scope of work and associated service levels takes a substantial effort; you should assume that negotiating the deal will take several months from RFP to contract for small or heavily fast-tracked deals, and even longer for large or unusually complicated deals.

By using an RFP, clearly documenting the scope, SLAs, and pricing, and making sure the deal is closed when it is signed, you can dramatically increase the chances that your outsourcing will work. If you involve an experienced deal consultant in the process, those odds will get even better.

an introduction to C#

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

C# (pronounced "cee sharp") is a new programming language, part of the .NET Framework initiative from Microsoft. This column is the first of a series that will discuss the C# language and libraries. But before we delve into technical details, we need to present a little background and show where C# fits into the larger picture.

A Proprietary Language?

An obvious question about C# is whether it is simply another in a series of proprietary languages (e.g., Visual Basic). Such languages are clearly useful but live in a different world from standardized languages like C. It's impossible to predict the future with any certainty, but C# does have a shot at becoming a widely used standard. The language has a specification external to Microsoft, and several independent projects are underway to develop C# compilers: for example, the Mono effort for Linux. We will give details of these efforts later in the discussion.

The .NET Framework

If you read the technical press at all, you've probably heard of something called the .NET Framework. This is an elusive term. What does it mean? One way of illustrating the concepts of the .NET Framework is to consider what happens when a C# program is compiled and executed. Let's start with the Hello program:

```
using System;

class Hello {
      static void Main() {
          Console.WriteLine("Hello, World!");
      }
}
```

The first interesting part of .NET is compilation. This program is compiled into an intermediate language called MSIL or IL, not straight into binary code. If I'd written the same program in a different language supported by .NET, the IL representation would be similar to what is produced for the C# program above. This point illustrates one of the key goals of the .NET effort – the ability to mix code written using different programming languages. This goal is supported by a common intermediate language.

The IL output for the Hello program looks like this:

```
.method private hidebysig static void Main() cil managed
{
      .entrypoint
      // Code size    11 (0xb)
      .maxstack 1
      IL_0000: ldstr "Hello, World!"
      IL_0005: call void [mscorlib]System.Console::
                  WriteLine(string)
      IL_000a: ret
} // end of method Hello::Main
```

The program is compiled into an intermediate language which is then executed at some later time. What happens then? A piece of the .NET Framework called the Common Language Runtime (CLR) actually executes the intermediate form of the program. The intermediate is compiled on demand into machine code, using a just-in-time compiler (JIT), and executed. The CLR also takes care of issues such as memory layout, garbage collection, and security. Furthermore, it provides a certain execution environment paradigm that supports the languages available for .NET.

If you study this example a bit, it's obvious that the code is making reference to some sort of a standard library – note the mention of System and Console.WriteLine, and so on. Another part of the .NET Framework is a set of framework base classes, used for performing operations such as I/O, string manipulation, and networking. The Hello program makes use of some of these classes for actual output to the console.

As we already mentioned, it's possible to mix languages and libraries within .NET. So in our Hello example, it's possible that the Console.WriteLine method for doing I/O is not in fact written in C# – it may be implemented in some other language. A couple of pieces of the .NET Framework called the Common Language Specification and Common Type System are used to support such interoperability. These specifications describe areas such as inheritance, object properties, exceptions, interfaces, and values. This whole area is in some sense analogous to the older terms "calling conventions" and "runtime environment" that we've always had to worry about when mixing languages. For example, if I have some C++ code, and I call a C function, compiled with a different compiler, then I need to

worry about such things as whether function arguments are pushed onto the stack from left to right or right to left. I need to be concerned with whether two different languages that I'm working with use the same byte order and size and representation for data.

Higher-Level Services

In discussing the .NET Framework, thus far we've looked at low-level features. There's the Common Language Runtime, along with standards such as the Common Type System and the Common Language Specification that describe how languages interoperate. There is also a set of framework base classes that is part of this core package.

The .NET Framework also contains several groups of higher-level services and classes:

- ADO.NET supports database manipulation and XML data handling.
- Windows Forms are the basic mechanism for building windows-based applications. A Forms object represents windows in your application.
- Web Forms are a mechanism for dynamically generating Web pages on a server, combining a static HTML page with C# code that generates dynamic content. The C# code runs on a server with the resulting generated HTML page being sent to a Web browser. Web Forms are something like Active Server Pages.
- Web Services allow you to build components whose methods can be invoked across the Internet. It is based on SOAP (Simple Object Access Protocol), which in turn is based on XML, HTTP, and SMTP.

You can also combine C# code with code written in other .NET languages, including VB.NET, Managed C++, and JScript .NET.

The .NET Framework Hierarchy

If we represent the .NET Framework using a hierarchy from highest to lowest levels, it would look something like this:

C#, VB.NET, Managed C++, JScript .NET
Windows Forms, Web Forms, Web Services
ADO.NET, XML
Framework Base Classes (I/O, string, networking, etc.)
Common Language Runtime
  Memory Layout
  Garbage Collection
  Security
  Debugging
  Exception Handling
  Just-in-Time Compilation
Common Type System, Common Language Specification
Operating System

The Flavor of C#

What is C# like? Let's look at a few key areas to help answer this question.

C# is an object-oriented language, similar in many ways to the C++ and Java languages. It also uses syntax similar to what you're already familiar with in C or C++. For example, this program adds two numbers and prints the sum:

```
using System;

class prog1 {
     static int add(int a, int b) {
        return a + b;
     }

     static void Main() {
        int a = 37;
        int b = 47;

        int c = add(a, b);

        Console.WriteLine("{0}", c);
     }
}
```

C# is a "safe" language, meaning that common problems such as memory leaks, de-referencing invalid pointers, or ignoring error return codes are much less of an issue than with other languages. C# uses garbage collection instead of user-level memory management, doesn't normally allow the use of pointers, and uses exceptions to propagate errors. If you need to use pointers, you can explicitly do so by means of an "unsafe" method modifier that allows pointers within that method. For example:

```
using System;

class prog2 {
     unsafe static void Main() {
        char* p = (char*)0x1234;
        *p = 'x';
     }
}
```

C# supports attributes and metadata and reflection. For example, you can devise custom attributes and use them to represent detailed information about bug fixes you have made in your code. Such information could also be represented within comments, which is a traditional approach, but attributes have a major advantage – they are not unstructured comments but can be queried by a C# program. They are data about your code that is carried along with your code.

C# is Internet-centric. For example, it includes support for remote method invocation, XML and XML documentation

comments, serializing objects to be sent across a network, and so on.

In this column we've described the context in which C# operates. In future columns we'll start looking at the language itself, and examine some of its distinctive features.

References

Many C# books are available. Two recommended ones are:

Jesse Liberty, *Programming C#*, 2d ed. (Sebastopol, CA: O'Reilly, 2002).

Eric Gunnerson, *A Programmer's Introduction to C#*, 2d ed. (Berkeley, CA: Apress, 2001).

C# Compilers and Development Environments

Here are Web links for three different C# compilers you can download. The first two of these are independent efforts, and the last is the Microsoft SDK:

http://www.go-mono.com/c-sharp.html

http://www.southern-storm.com.au/portable_net.html

http://msdn.microsoft.com/downloads/default.asp?URL=/code/sample.asp?url=/msdn-files/027/000/976/msdncompositedoc.xml

C# Standardization

The C# language has an external specification, found at the European Computer Manufacturers Association (ECMA) Web site: *ftp://ftp.ecma.ch/ecma-st/Ecma-334.pdf*.

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

Analyzing Network Usage

The previous Tclsh Spot article (October 2002 *;login:*) described using SWIG to build a Tcl extension that could build and transmit datagrams over an Ethernet.

Before I'd finished the article, a friend requested a package that could transmit various datagrams over an Ethernet, but it needed to be fast. He was concerned that an interpreted language like Tcl wouldn't be able to put datagrams onto the Net fast enough.

When I test a system, I prefer to use some other platform to test it from. Since I'm generating the packets with a Tcl script on one computer, I prefer to analyze the output on another piece of hardware.

Fortunately, I have a Spirent/AdTech AX-4000 broadband analyzer handy, and it can be programmed using Tcl.

This article will briefly describe the Spirent/AdTech AX-4000, and the AdTech Tcl extension, and show how to use the equipment to check how fully a network is being utilized.

The AX-4000 (*http://www.adtech-inc.com/*) is a configurable piece of hardware that can generate and analyze data packets on four different transmission technologies (IP, ATM, Ethernet, and Frame Relay) simultaneously at speeds up to 10Gbps.

A simple system will include a controller card set and an interface card set. The interface card set contains the circuitry to generate and analyze packets for a transmission medium. A controller card set can control multiple interface cards, which can be mixed and matched to work with Ethernet, fiber, etc.

For this example, the AX-4000 is equipped with a controller and an Ethernet interface card set.

The AX-4000 comes with a nice GUI for performing bench testing, and it also includes a Tcl extension. The primary purpose for the Tcl extension is to support automated testing, but it's also useful for folks who prefer to work outside the GUI.

The general flow for an AdTech Tcl script is:

1. Load the AxTcl extension.
2. Initialize the connection to the AX-4000 controller.
3. Reserve an interface card set.
4. Create a generic interface object attached to the card set.
5. Create an analyzer or generator attached to the interface.
6. Configure the analyzer or generator.
7. Run the test.
8. Analyze the results.

Compiled Tcl extensions can be loaded with the Tcl load command:

Syntax: load *libFile.so ?name?*

Load a shared library extension into the Tcl interpreter.

libFile.so The name of the library to load. The filename suffix will depend on the base operating system.

?name? An optional name for the Tcl initialization function.

The AxTcl extension is available for Solaris, Linux, or Windows platforms. One trick for writing a script that will load on all platforms is to use the catch command to see if the extension loads correctly, and step on to the next possibility if the load fails.

```
if {[catch {load $base/tclwin/libax4k.dll ax4kpkg}]} {
    catch {load $base/tclclib/libax4k.so ax4kpkg}
}
```

Once the AxTcl extension is loaded, it creates several new Tcl commands, each of which has several subcommands. The new commands include:

ax	Interacts with an AX-4000 system.
interface	Establishes and configures a connection to the generic interface.
enet	Interacts with an Ethernet connection.
analyzer	Establishes and configures a connection to the analyzer.

The ax commands provide the high-level control needed for the interactions with the AX-4000 equipment.

One of the features that make the AX-4000 series so fast is that they make heavy use of programmable logic. This feature allows

the hardware to be configured for specific tasks, which are then hardware driven rather than software driven. Programming the hardware allows the AX-4000 to do things like saturate the largest optical fiber with packets and analyze them in real time.

The programmability of the AdTech hardware also means that special hardware configuration files must be available to program the boards for the various tests to be performed.

The ax hwdir command tells the AxTcl extension where to find the hardware configuration files.

Syntax: ax hwdir *path*

ax hwdir *directory*	Identifies the directory for the AX-4000 BIOS files. Default is: ../bios

Once the system knows where to find the BIOS files with the programmed logic definitions, you can initialize the connection to the AX-4000 with the ax init command.

Syntax: ax init *?-option value?*

Initialize internal tables in the AX library.
Options include:

-remote *IP*	The IP address of an AX-4000 accessed via an Ethernet port.
-user *name*	A username that will be used to identify who is using this AX-4000.
-nobios *1/0*	By default ax init will download BIOS to a freshly powered-on AX-4000. Setting this to 1 will inhibit that download.
-forceload *1/0*	Forces the AX-4000 to get a new BIOS upload. When working with multiple revisions of AxTcl, this is recommended.

Initializing the connection to an AX-4000 can be done with these two lines of Tcl code:

```
ax hwdir $base/bios

ax init -remote $ipAddress -user clif -forceload 0
```

The AX-4000 can support multiple users and multiple interface cards on a chassis, but only one user at a time can use an interface card set. To avoid having two applications fighting for control of a card set, the AxTcl extension allows an application to lock (and release) the physical card set for an application's use.

The two commands that control this for an Ethernet card set are enet lock and enet unlock.

Syntax: enet lock *LogicalID ControllerIndex DeviceID*

	Locks a device for this script's use and assigns a logical ID to that device. NOTE: Throws an error if device is already locked.
LogicalID	A value provided by the script to use to reference this locked device.
ControllerIndex	The IP Address/Hostname of this AX-4000.
DeviceID	The position of the card being locked (counting from 1).

Syntax: enet unlock *LogicalID*

enet unlock	Unlocks a device identified by LogicalID from a previous enet lock command.
LogicalID	The device identifier assigned in a previous enet lock command. If this parameter is left out, all devices previously locked in this session are unlocked.

The enet lock command will throw an error if another user has locked a card set. Once the lock has been successful, however, a script can create an interface to the card set.

Syntax: interface create *Name Device ?-key value?*

interface create	Create a new interface object.
Name	The name to assign to the new interface.

Device	The device to attach this interface name to.
?-key value?	Option and value pairs to control how the interface behaves or to configure the card set.

These options vary from card set to card set, and may include:

-interface A|B	For dual interfaces, selects the left (default) as A or right B interface.
-ifmode *type*	Defines the type of data to be used on this interface. Values for interface type include:
POS	Sonet Packets.
IPoETHER	Internet protocol datagrams encapsulated in Ethernet frames.
IPoPPP	Internet protocol datagrams encapsulated in PPP frames.
IPoATM	Internet protocol datagrams encapsulated in ATM frames.
IPoFR	Internet protocol datagrams encapsulated in Frame Relay frames.

The interface create command creates a new command with the same name as the interface you've created. Your script will use this new command to interact with the interface. Two of the main subcommands for the new interface are set (to set device-specific options) and run (to start the interface).

The code to create, configure, and start an interface looks like this:

```
interface create int1 $logicalID -ifmode IPoETHER
int1 set -mode normal -dataRate MBS10
int1 run
```

The next step is to create an analyzer and/or generator object attached to the interface object. Creating the analyzer or generator follows the same pattern as creating the interface.

Syntax: analyzer create *Name Device*

analyzer create	Create a new analyzer object.
Name	The name to assign to the new analyzer.
Device	The device to attach this analyzer name to. This is the logical device that was locked in a previous enet lock command.

The analyzer create command will create a new analyzer object and a new command to use to interact with that object. The analyzer command supports many subcommands, including:

analyzerName set *Name Device*
: Sets one or more configuration options for this analyzer. Configuration options vary for different analyzer cards.

analyzerName display
: Returns a list of the current settings.

analyzerName run
: Starts the analyzer running.

analyzerName reset
: Stops the analyzer and clears all the statistics the analyzer can gather.

analyzerName stop
: Stops the analyzer but does not clear any values.

analyzerName destroy
: Destroys the analyzer, freeing it for other use.

analyzerName stats
: Returns a set of keyword-value pairs as a list. The exact return depends on the analyzer being used.

The analyzer can do lots of interesting things, including capturing packets, generating histograms of the data, and much more. For this application, all we need is to look at the statistics that the AX-4000 analyzer gathers whenever it's running.

This code resets the analyzer, runs it for two seconds, collects the runtime statistics, and releases the device for other users.

```
# Reset the statistics to 0
ana1 reset.
```

```
# Pause until the AX-4000 completes its action
after 400
# Start the analyzer
ana1 run
# Wait 2 seconds and get the statistics
after 2000
set anaStats [ana1 stats]
# Stop the hardware and
# destroy the software object
ana1 stop
ana1 destroy
# Finally, unlock the device for the next user
enet unlock $logicalID
```

Most AxTcl commands return their results as a list of keyword and value pairs. The Tcl foreach command makes this data format easy to use.

Syntax: foreach *varList dataList body*

	Evaluate *body* for each of the items in *dataList*.
varList	A list of variable names. Data values will be extracted from the *dataList* and assigned to these variables.
dataList	A list of data values to step through.
body	The body of code to evaluate on each pass through the loop$

The data will be easier to read if it's formatted as columns. The Tcl format command implements the same string formatting rules as the C library sprintf command.

Syntax: format *formatString value1 ?value2?...*

This code will display a table of keywords and values from the analyzer:

```
puts "ANALYZER STATS"
foreach {key1 val1} $anaStats {
    puts [format "%-30s %12s" $key1 $val1]
}
```

The output looks like this:

```
ANALYZER STATS
-elapsedTime                  2081
-totalPackets                19110
-totalPacketBytes          1949220
-goodPackets                 19110
-goodPacketBytes           1949220
-goodDatagramBytes         1605240
-totalPacketRate             10160
-goodPacketRate              10160
-goodPacketBitRate            8291
-goodDatagramBitRate        6827.5
-lineRatePerc                111.70
-tcpPackets                      0
-tcpRatio                     0.00
-tcpChecksumErrors               0
-udpPackets                      0
-udpRatio                     0.00
-udpChecksumErrors               0
-icmpPackets                19110
-icmpRatio                    1.00
-ipPackets                 19110.00
-ipChecksumErrors             0.00
-avgDatagramLength              84
-minDatagramLength              84
-maxDatagramLength              84
-avgPacketLength               102
-minPacketLength               102
-maxPacketLength               102
-substreamCount                  1
-substreamErrorCount             0
-filterCount                     2
```

Dividing the -totalPacketBytes value (1,949,220) by the 2.081 seconds that the test ran gives 936,674 bytes/second, which is fairly close to 100 percent usage of the 10 megabit/second theoretical bandwidth of the network.

This provides a quick introduction to the AX-4000 and AxTcl. The next few articles will discuss generating different types of Ethernet frames, verifying the generator with the AX-4000, and using those frames to validate a Linux-based firewall.

As usual, the code for these examples is available at *http://www.noucorp.com*.

practical perl

CPAN, Modules, and the CPAN Shell

by Adam Turoff

Adam is a consultant who specializes in using Perl to manage big data. He is a long- time Perl Monger, a technical editor for *The Perl Review*, and a frequent presenter at Perl conferences.

ziggy@panix.com

If you asked a room full of Perl programmers to name their favorite feature of Perl, the vast majority of them would say that it is CPAN, the Comprehensive Perl Archive Network. Many non-Perl programmers agree that CPAN is the most interesting feature about Perl.

CPAN is a globally distributed library of scripts, modules, documentation, and other resources created by and for Perl programmers. If you need to create an application with a graphical user interface, connect to a database, or access a Web service, you can find a great many modules to help you at your local CPAN mirror. Currently, over 200 public CPAN mirrors, plus countless private mirrors, are available around the world.

Most Perl programmers are familiar with the two primary Web-based interfaces to CPAN: the main Web page, *http://www.cpan.org/* (also available at your local CPAN mirror), and the CPAN search engine, *http://search.cpan.org/*. Both of these sites are excellent resources if you are looking for Perl modules to use. However, finding a module that may help you solve a problem easily is only half of the battle. Fortunately, searching is the difficult part; installing a module once you know its name is much much easier, thanks to the CPAN shell.

The CPAN Shell

One of the many core modules that comes with every version of Perl released since 1997 is Andreas Koenig's CPAN module. This very useful module has a great many features, but its overall purpose is to help you maintain a Perl installation by making it easy to upgrade and install Perl modules from your local CPAN mirror.

The most common way to use the CPAN module is to use the CPAN shell:

```
[ziggy@chimay ~]$ perl -MCPAN -e shell

cpan shell – CPAN exploration and modules installation (v1.63)
ReadLine support enabled

cpan>
```

Within the CPAN shell, installing modules is easy – just type install and a list of modules to install. The install will download, extract, configure, build, test, and install a module.

```
cpan> install LWP::Simple
....
cpan> install Bundle::DBI DBD::mysql DBD::SQLite
....
```

Another way to use the CPAN module is to install modules directly from the command line:

```
[ziggy@chimay ~]$ perl -MCPAN -e 'install LWP::Simple'
....
[ziggy@chimay ~]$
```

The CPAN shell has many other uses. It can act as a basic search tool, report on what modules you have installed, and determine which installed modules are out-of-date. To inspect a module, just type m *modulename* and it will display the current version and, if it is installed, the current location of that module:

```
cpan> m DBI
Module id = DBI
    DESCRIPTION     Generic Database Interface
                    (see DBD modules)
    CPAN_USERID     TIMB (Tim Bunce
                    <dbi-users@perl.org>)
    CPAN_VERSION    1.32
    CPAN_FILE       T/TI/TIMB/DBI-1.32.tar.gz
    DSLI_STATUS     MmcO (mature,mailing-list,
                    C,object-oriented)
    MANPAGE         DBI - Database-independent
                    interface for Perl
    INST_FILE       /opt/perl/lib/site_perl/5.6.1/
                    i386-freebsd/DBI.pm
    INST_VERSION    1.30

cpan> m DBD::Oracle
Module id = DBD::Oracle
    DESCRIPTION     Oracle Driver for DBI
    CPAN_USERID     TIMB(Tim Bunce
                    <dbi-users@perl.org>)
    CPAN_VERSION    1.12
    CPAN_FILE       T/TI/TIMB/DBD-Oracle-1.12.tar.gz
    DSLI_STATUS     MmcO (mature,mailing-list,C,
                    object-oriented)
    INST_FILE       (not installed)

cpan>
```

To search for modules, use the m command with a regular expression to find a list of matching module names:

```
cpan> m /^Sort/
Module  Sort::ArbBiLex      (S/SB/SBURKE/
                            Sort-ArbBiLex-3.4.tar.gz)
```

```
Module  Sort::Array          (M/MI/MIDI/
                             Sort-Array-0.26.tar.gz)
Module  Sort::ArrayOfArrays (E/EA/EARL/Sort-
                             ArrayOfArrays-1.00.tar.gz)
Module  Sort::Fields         (J/JN/JNH/
                             Sort-Fields-0.90.tar.gz)
Module  Sort::Naturally      (S/SB/SBURKE/
                             Sort-Naturally-1.01.tar.gz)
Module  Sort::PolySort       (Contact Author
                             Daniel Macks
                             <dmacks@netspace.org>)
Module  Sort::Versions       (E/ED/EDAVIS/
                             Sort-Versions-1.4.tar.gz)
Module  sort                 (J/JH/JHI/perl-5.8.0.tar.gz)
8 items found

cpan>
```

The CPAN shell has a great many other uses. You can find more documentation bundled with the CPAN module by using perldoc CPAN or man CPAN.

Configuring the CPAN Shell

The first time you run the CPAN shell, it will take you through a quick configuration process, which starts out like this:

```
[ziggy@chimay ~]$ perl -MCPAN -e shell

/home/ziggy/.cpan/CPAN/MyConfig.pm initialized.

CPAN is the world-wide archive of perl resources. It
consists of about 100 sites that all replicate the same
contents all around the globe. Many countries have at
least one CPAN site already. The resources found on
CPAN are easily accessible with the CPAN.pm module. If
you want to use CPAN.pm, you have to configure it
properly.

If you do not want to enter a dialog now, you can answer
'no' to this question and I'll try to autoconfigure. (Note:
you can revisit this dialog anytime later by typing 'o conf
init' at the cpan prompt.)

Are you ready for manual configuration? [yes]
```

I have found that the defaults are quite sensible. To save time, you can answer "no" to this prompt and accept all default values. The one value that cannot be intuited is the URL for your local CPAN mirror. If you do not know where to find your local CPAN mirror, then it is best to go through the manual configuration. You will then be presented a list of CPAN sites that are close to you geographically. The manual configuration is also a good idea if you need to use an FTP or HTTP proxy to connect to a CPAN mirror.

If you do know the URL of a local CPAN mirror, you can accept all of the other defaults (by typing "no" at the manual configuration prompt) and specifying it directly. This can be done with the o conf urllist command:

```
[ziggy@chimay ~]$ perl -MCPAN -e shell

cpan shell – CPAN exploration and modules installation (v1.63)
ReadLine support enabled

cpan> o conf urllist push http://www.cpan.org/

cpan>
```

You can see all of the options used by the CPAN shell through the o conf command. When you change the value of an option, it will be used for the duration of your shell session. To save these values permanently, use the o conf commit command:

```
cpan> o conf commit
commit: wrote /home/ziggy/.cpan/CPAN/MyConfig.pm

cpan>
```

Maintaining Multiple Perls

The CPAN module can be configured in two basic modes. When run as root, the default CPAN configuration will be site-wide and stored globally in the CPAN::Config module. Alternatively, you can use the CPAN shell as an unprivileged user, and the CPAN configuration will be stored as CPAN::MyConfig in your home directory (~/.cpan/CPAN/MyConfig.pm to be precise).

On my personal machines, I tend to have multiple versions of Perl installed. First is the version of Perl that comes with the operating system: FreeBSD 4.x ships with Perl 5.005_03 (released March 28, 1999), and MacOS X ships with Perl 5.6.0 (released March 23, 2000). Both of these versions have been superseded by Perl 5.6.1 (released April 9, 2001). I do most of my work with Perl 5.6.1 and do my best to leave the vendor-installed version of Perl alone.

Last summer, Jarkko Hietaniemi and the perl5-porters released Perl 5.8.0, a very major upgrade that includes many new and improved features. I am currently playing with some of these new features, and maintain this installation alongside my installation of 5.6.1. Keeping multiple releases of Perl around helps when I test to see if my programs will work with older versions.

I have also found that upgrading Perl versions where critical production programs are in use can be problematic. The last thing anyone wants to do is break a critical program by upgrading one of its dependencies (like Perl or some Perl modules). Unfortunately, this means that many developers are constrained to write and deploy software using an older version of Perl. Keeping multiple versions of Perl installed is one way to let programmers use the newer features available in newer releases of Perl when writing new programs without interfering with the

critical programs that are best left alone. This strategy also makes it easier to slowly migrate critical programs to newer versions of Perl in a controlled manner, or test a program against multiple releases of Perl.

Because I keep multiple versions of Perl on the same machine, I want to avoid using the CPAN shell as root. When the CPAN module is configured globally, each specific version of Perl must be configured individually. That is because the configuration stored in CPAN::Config is site-wide, but only for a specific Perl installation, so five Perl installations means that five CPAN::Config files need to be created and maintained.

The alternative is to use a user-level configuration, where the configuration is stored in my home directory. That way, CPAN::MyConfig can be configured once, and that configuration will be used by the CPAN shell with all versions of Perl I have installed:

```
[ziggy@chimay ~]$ /opt/perl/bin/perl5.6.1 -MCPAN -e shell
....

[ziggy@chimay ~]$ /opt/perl/bin/perl5.8.0 -MCPAN -e shell
....
```

Using the CPAN shell as an unprivileged user raises a minor issue. While I can download, extract, configure, build, and test a module, I cannot install it in the system-wide library. To do that, I need to have superuser permissions, just as I would with any other software install. This problem is easily solved by using sudo to run the CPAN shell. A judicious use of sudo can let multiple users maintain Perl installations on a single system, or expressly specify which users can maintain which specific Perl installations.

Testing Modules Locally

By default, the CPAN shell will aid the process in adding modules to the local site library. Usually, this is what you want to do.

What do you do when it is time to upgrade a critical module? Do you take the chance that the upgrade will not break anything, or do you test it first? There are many ways to solve this problem. The most troublesome and labor-intensive solution is to maintain a separate test machine for testing new Perl modules before they are deemed ready to install. A better alternative is to maintain a secondary "scratch" installation of Perl where modules can be installed and tested. This is very easy to do, requiring that one version of Perl be installed in two or more separate locations.

The easiest solution is to avoid the problem of updating the site-wide library for a Perl installation and just test modules in a local library area. To do this, I start by creating a "test" account that has very limited access to the system (no group memberships, no sudo access). I then set up a CPAN shell configuration specific to this user that will install all CPAN modules in the /home/test/lib directory. Finally, I tell Perl to look in this directory *before* looking in the site-wide module library areas, so that I can install and test both new modules as well as upgraded versions of previously installed modules.

Of course, there are many ways to tell Perl where to look for modules. The list of directories that contain Perl modules is stored in @INC. Here is one common idiom for adding a directory to the list of directories to search:

```
#!/usr/bin/perl -w

BEGIN {
    unshift(@INC, "/home/test/lib");
}

use strict;
....
```

That technique is rather opaque. Recent versions of Perl now include a use lib pragma to specify an alternate module directory. Using this pragma is preferable to using the old style BEGIN block:

```
#!/usr/bin/perl -w

use strict;
use lib '/home/test/lib';
....
```

If I can modify programs I want to test, the use lib technique will work. If I want to test a program that I cannot modify, or do not wish to modify, I can specify additional library paths when I invoke Perl. One approach is using the PERL5LIB environment variable. Another approach is to use the -I command line switch. I can see impact on the module search path (stored in @INC) simply by printing it out:

```
[test@chimay ~]$ perl   -I/home/test/lib \
>                       -e 'print join("\n", @INC), "\n"'
/home/test/lib/5.6.1/i386-freebsd
/home/test/lib/5.6.1
/home/test/lib/
/opt/perl/lib/5.6.1/i386-freebsd
/opt/perl/lib/5.6.1
/opt/perl/lib/site_perl/5.6.1/i386-freebsd
/opt/perl/lib/site_perl/5.6.1
/opt/perl/lib/site_perl
.
[test@chimay ~]$ PERL5LIB=/home/test/lib \
>                perl -e 'print join("\n", @INC), "\n"'
/home/test/lib/5.6.1/i386-freebsd
/home/test/lib/5.6.1
/home/test/lib/
/opt/perl/lib/5.6.1/i386-freebsd
```

```
/opt/perl/lib/5.6.1
/opt/perl/lib/site_perl/5.6.1/i386-freebsd
/opt/perl/lib/site_perl/5.6.1
/opt/perl/lib/site_perl
.
[test@chimay ~]$
```

Updating the CPAN Configuration

The next thing I need to do is update the CPAN configuration for the "test" user so that it installs modules in /home/test/lib. The best way to specify this is to use the PERL5LIB prefix, because the configure/build process will invoke some Perl subprocesses. If I specify the library path using -I, then the CPAN shell will find modules installed there, but the subprocesses will not. Using the PERL5LIB environment variable fixes this problem.

I start by logging in as "test" and invoking the CPAN shell so that it can find my local module directory:

```
[test@chimay ~]$ PERL5LIB=/home/test/lib perl -MCPAN -e shell
cpan>
```

The vast majority of Perl module distributions are built so that they can be installed easily with CPAN.pm. The process is quite simple (and easily automated by the make and install commands). The configuration process (perl Makefile.PL) generates a makefile that will be used to build and install a module. This auto-generated makefile provides many options for overriding the default configuration parameters.

One of the makefile parameters that can be configured is the PREFIX variable, which defines the root directory of a Perl installation. By setting this variable to /home/test when generating the makefile, the install process will install Perl modules under /home/test/lib, man pages under /home/test/man, and programs under /home/test/bin.

Another parameter that can be configured is the INSTALLDIRS variable. This specifies one of three possible areas where modules can be installed: perl, site, or vendor. In a stock Perl configuration, modules installed under the perl directory go into /usr/local/lib, modules installed under the site directory go into /usr/local/lib/site_perl, and the vendor directory is unused.

Because we may be installing upgrades to Perl core modules, it's important to set the INSTALLDIRS variable when generating a makefile so that all modules go into the same local directory. For convenience, I set this variable to perl, so that all modules will be installed in /home/test/lib instead of /home/test/lib/. Updating the makepl_arg option to specify these two configuration parameters is simple:

```
cpan> o conf makepl_arg "PREFIX=/home/test
                                INSTALLDIRS=perl"
      makepl_arg     PREFIX=/home/test INSTALLDIRS=perl

cpan> o conf commit
commit: wrote /home/test/.cpan/CPAN/MyConfig.pm

cpan>
```

Now, the CPAN shell is configured to find modules in /home/test/lib and install new modules in that location.

Any instance of this version of Perl that starts up with -I/home/test/lib specified (or PERL5LIB=/home/test/lib) will use the modules installed in this location. Alternatively, any program that contains a use lib '/home/test/lib'; declaration will find the upgraded version of the CGI module. All other programs will find the previously installed version.

Conclusion

The CPAN shell is a very useful tool for installing Perl modules on your system. Although it is typically used for managing one Perl installation on a system, it can be used to help maintain multiple parallel Perl installations. The CPAN shell can also be used to aid in testing and evaluating Perl modules before installing them system-wide.

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

It's the dead of winter, and global warming seems more like a mirage than the reality that it is. I can see snow on the Mogollon Rim, the southwest edge of the Colorado plateau, a welcome sign of moisture that will hopefully be the end of four years of drought.

On that cheery note, I get to muse about security, the topic that makes me pointy-headed. And a couple of things have got me riled up, both as a result of attending yet another security conference.

While teaching, I covered the topic of tunneling through firewalls. Tunneling has become very popular in open source of late, since four distributions had been trojaned as of the end of November 2002. In each case, someone broke into an FTP site and modified the configure script for a particular package so that it would compile and execute an additional program. This program, the trojan, runs in the background with the privileges of the user who executed configure – another reminder of why you avoid doing anything other than what you must do as root. And what the trojan program does is pretty cute.

Once an hour, the trojan initiates an outgoing connection to a fixed IP address, and port 6667/TCP. If the connection is successful, it only remains open for 10 seconds, unless some input is received. When something is received, the trojan executes a shell. I suppose that the attackers had arranged something akin to an expect script at the remote end of this connection, which will download assorted tools and kits, and automatically continue to exploit the victim of the trojan. Of course, as the trojan used fixed IP addresses (different for each instance), as soon as someone noticed the attack, the site receiving the connections could be taken off the Net and cleaned up.

Once upon a time, I used to counsel people to examine the source code that they download from the Internet before using it. Today, that admonition is incredibly absurd, especially when you can download entire CDs containing operating systems along with software packages numbering in the tens of thousands. I recently grabbed a Linux rootkit, to use in a class example, and realized that checking it out for *unexpected* back doors (I knew about the well-known ones) was going to take longer than I cared to spend. There are shortcuts, such as looking for system calls that open files, create sockets, and execute programs. The trojan mentioned above both creates sockets and executes a shell, so it qualifies. But who even considers checking for these things? Keep in mind that many programs do this legitimately. The configure script found in cfengine (not one of the victims) contains 48 lines that include `socket`. Checking the GPG signature of packages at least assures that nothing has been added since the package was signed.

Sysadmins working at sites with serious firewalls can consider blocking all outgoing traffic except that which is permitted and expected. Blocking arbitrary outgoing traffic, and watching your logs for any deny messages actually will help you discover exploited systems on your internal networks.

That is, unless the attacker is using HTTP over port 80/TCP. If someone out there works at a site connected to the Internet that does not permit access to most Web servers, please let me know. I include the word "most" for those sites unlucky enough to be behind firewalls that block requests for URIs by using "evil"-word or other filtering mechanisms. I really wonder how well such things can work, after winding up at a pornography page that had replaced Lord Somer's Lurker rootkit site.

If the writer of the configure trojan had decided to make his (or her) program about twice as long, it could have used port 80/TCP and included fake headers on the requests, and stripped off the server headers from the replies. Doing so might have made the trojan easier to spot (it is actually well disguised, complete with comments that help it blend into any configure script). But it would have made it much more difficult to prevent it from making outgoing connections through firewalls that permit HTTP.

And it is not only this trojan that uses this technique. An IIS 5 exploit named "jill" also made outgoing connections but, in this case, using the IP address and port of the attacker's choice. You should consider exploits that make outgoing connections something that you should expect, and not something rare and unusual.

SOAP

Exploits won't be the only thing slipping through your firewall soon. SOAP, the Simple Object Access Protocol (*http://soap.weblogs.com/*), also uses HTTP, with the actual goal being able to penetrate firewalls at will. SOAP carries XML payloads used to invoke remote methods, carry executable code, and return responses. Now, you can already invoke remote methods using CGI, so that is really nothing new. And returning results, ho-hum. But carrying remote code, well now, I hope that got your attention.

SOAP has become a carrier for .NET, Microsoft's new programming paradigm. DCOM, the old paradigm, did support the downloading of remote code and its execution using plain old HTTP. DCOM has, as one of its drawbacks, no means for running remote code securely. The .NET framework gets around this by permitting the programmer to request only the privileges needed to execute on the victim's, er, remote user's system. And the CLR (Common Language Runtime) that will execute the MSIL (Microsoft Intermediate Language) module that has been downloaded can also set limits on the privileges allowed the code, based on the source of the code and its authenticating signature.

VB.Net and C# both produce MSIL, something that is vaguely like Java bytecode. But, MSIL does not include the same security paradigms as Java, which means that programmers can still make mistakes that would be impossible using Java bytecode. Also, there is no security provided by SOAP or XML – they just carry the code. In other words, .NET application security depends entirely on the security skills of the programmer writing the application server or remote modules. Does this sound familiar?

If you want to get an idea about how things can go wrong, just check out MS-02-065. All MDAC (an ActiveX module used by IIS and IE) versions until 2.7 (the one installed on XP) have a buffer overflow that can be exploited using Web accesses (if RDS is enabled), HTML, and email that includes HTML. The MDAC ActiveX module has been digitally signed by Microsoft, so even if you never had it, or had removed it, a malicious email could be used to load a vulnerable version, and it would be trusted – because it was signed by Microsoft. You could actually disable Active Scripting (a very good idea, suggested by Microsoft), as well as remove Microsoft from the list of trusted providers of code (*http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS02-065.asp*).

SOAP appears to be an enabling technology for security holes. It supports transmission of mobile code while making it very difficult to check and see what is being sent – that information is buried in the XML. It really is a shame that security isn't the first consideration when new protocols are designed, instead of an afterthought. And on that note, I wish you a warm good night.

.NET application security depends entirely on the security skills of the programmer writing the application server or remote modules.

web browser vulnerabilities 101

by Peleus G. Uhley

Peleus Uhley is the senior developer for Anonymizer Inc., where he is responsible for the privacy surfing service and privacy analysis "Snoop" page.

peleus@anonymizer.com

Recently, Microsoft Internet Explorer made the SANS/FBI Top 20 Security Vulnerabilities list.[1] For followers of BugTraq, you will have seen new postings of browser vulnerabilities monthly over the past year. For all the hype surrounding these issues, how is the browser a vulnerability? If you don't visit hacker sites, is there a threat? The answer is, sadly, yes in more instances then you might expect. This article focuses on Internet Explorer, but most of what is presented is true for any Web browser currently available.

To start, let's look at what it means to be a browser. Most people will answer with the most popular function, which is to transform the Hyper-Text Markup Language (HTML) into a viewable Web page. In the case of Internet Explorer, the browser can also interpret Java, ActiveX, JavaScript/JScript, VBScript, XML, XLST, and several other languages. Depending on the language, they may be compiled by the browser locally on the PC. The browser can launch almost any application, including media players and mail clients.

Internet Explorer is designed with the Microsoft Container-Object model, enabling you to view Word, Excel, and many other documents from within the IE container. The browser code overlaps with Windows' Explorer to access files on the Internet, in your network, and in your local file system. The browser can both send and receive files from the Internet. In addition, programs such as Outlook, Outlook Express, AOL, and MSN use the browser's internal engine to render HTML formatted email. The browser can use active content to have bi-directional communication between third-party software and itself.

Once you realize the full power of the browser, it becomes more apparent why it is such a targeted piece of software – it is the next best thing to hacking the OS itself! A recent WebSideStory report stated Internet Explorer is used by 95.97% of all Internet users.[2] The rate at which vulnerabilities are posted makes it very difficult for administrators and the general public to keep the browser patched at all times. In addition, Microsoft has not patched all the holes found within the browser! Although Microsoft was able to shorten the list of unpatched vulnerabilities from 31 down to 19 between November and December,[3] this has been a race they have been losing all year.[4]Add to this the fact that business and personal firewalls usually allow all outgoing port 80 traffic and you have a potentially high-risk situation for the personal workstation.

Types of Vulnerabilities

Vulnerabilities in Web browsers take many forms. The SANS/FBI report warns:

"The vulnerabilities can be categorized into multiple classes including Web page spoofing, ActiveX control vulnerabilities, Active scripting vulnerabilities, MIME-type and content-type misinterpretation and buffer overflows. The consequences may include disclosure of cookies, local files or data, execution of local programs, download and execution of arbitrary code or complete takeover of the vulnerable system."[5]

To give some definition to their classes and add examples, I provide the following general explanations with footnote references to specific examples.

Web page spoofing: In Web page spoofing the attacker makes you believe you are at a "safe" site when you are really at a site controlled by the hacker. These attacks can include altering IE's location bar to show the wrong URL, mixing real site content with

1. *http://www.sans.org/top20/* – SANS/FBI Top 20.
2. *http://www.Websidestory.com/cgi-bin/wss.cgi?corporate&news&press_1_193* – WebSideStory report on browser usage.
3. *http://www.pivx.com/larholm/unpatched/* – List of unpatched vulnerabilities.
4. *http://www.pivx.com/larholm/unpatched/archivednews.html* – History of IE vulnerabilities.
5. *http://www.sans.org/top20/#W8* — IE section of SANS/FBI Top 20.

altered content, and showing the title of the page being spoofed, making it almost impossible to determine that you are not where you think you are.[6]

ActiveX control vulnerabilities: Signed ActiveX controls run as resident programs on your PC with full privileges when loaded through IE. The operating system treats signed code as local code. By default, IE does not prompt the user about this action so long as the code is signed. If someone has access to a certificate, then this type of attack could be very transparent. For example, a malicious hacker could use this in order to load buggy DLLs signed by the original vendor to temporarily downgrade your computer.[7]

Active Scripting vulnerabilities: Although almost all of the attacks described use Active Scripting to operate; the scripting languages themselves can have vulnerabilities in their implementation within the browser. These usually lead to bypassing the Security Zone restrictions for local file access, program execution or Cross-Site Scripting (CSS) attacks.[8] Cross-Site Scripting is the ability for one site to gain access to another site's data such as their cookies or form information. Active Scripting attacks can involve JavaScript, JScript, VBScript, and XLSA. JavaScript is usually the language-of-choice for exploits.

Mime-type/Content-type vulnerabilities: Here the attacker falsely sends an incorrect file type in the headers to fool the user into downloading an executable. This could also be used to launch another application, such as a mail program to parse and run the active content outside of IE's restrictions.[9]

Buffer overflows: IE is just as vulnerable as the next program to this classic programming error. For IE, these can be infinite loops that crash the browser[10] or they can also be variable overflows.[110]

There are many ways to fool the user without using browser exploits. The Cuartango Window is the oldest example: Here a window with a harmless question such as, "Do you like chocolate?" covers a security window asking to run harmful code.[12] Users believe they are answering the chocolate question but the OS takes "yes" as the answer to whether to run the code. Another attack is to spoof the entire screen so that the user is no longer interacting with the OS![13]

Other recent problems include IE's SSL implementation allowing forged SSL connections through certificate chaining.[14] Some of Internet Explorer's default settings can also be a danger. By default, IE allows Web sites complete access to whatever information is currently copied onto your clipboard. In addition, many other programs interact with IE allowing for an attack through those programs.[15] There have been recent attacks against both Java[16] and IE's[17] compilers.

Almost all vulnerabilities reported to BugTraq include sample code (often only a few simple lines) making these attacks easy to implement.

Type of Threats

Some people may consider this a somewhat apocalyptic view. What if they only visit "safe" sites such as business and news sites? Most exploits require a person to visit an unsafe site to launch the exploit, so where is the problem? Browsers aren't always on servers and aren't always installed on servers so is there a corporate threat?

First of all, you have to remember that your Outlook mail clients and other pieces of software use IE's engine for rendering HTML-based email or connecting to the Net.

6. *http://www.securitytracker.com/alerts/2001/Dec/1003024.html* – Web spoof.
7. *http://www.guninski.com/signedactivex2.html* – Signed ActiveX of old DLLs.
8. *http://security.greymagic.com/adv/gm010-ie/* – "Who Framed Internet Explorer?"; *http://security.greymagic.com/adv/gm012-ie/* – "Vulnerable Cached Objects in IE (9 Advisories in 1)."
9. *http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms01-020.asp* – Microsoft's posting regarding MIME vulnerabilities.
10. *http://online.securityfocus.com/archive/1/269241* – Looped buffer overflow.
11. *http://online.securityfocus.com/archive/1/289106* – Buffer overflow to code execution.
12. *http://www.safenetworks.com/Windows/ie26.html* – Cuartango Window.
13. *http://www.guninski.com/popspoof.html* – Spoofing entire screen.
14. *http://online.securityfocus.com/archive/1/292842* – IE6 SSL certificate chain.
15. *http://online.securityfocus.com/archive/1/282631* – ICQ and MSIE.
16. *http://lists.netsys.com/pipermail/full-disclosure/2002-November/002730.html* - IE Java Vulnerabilities.
17. *http://online.securityfocus.com/archive/1/301220* - Netscape Java Vulnerabilities

Internet Explorer is "an integrated part of the Microsoft operating system." Patching your browser can help reduce the risk of viruses and attacks through these other programs.

On the Internet, you would most likely have to visit an unsafe site. However, the line between safe and unsafe is getting more and more blurry. Recently, spam that tells users they have an e-greeting card has been used to lure people into visiting a Web site and having them accept an ActiveX control to send spam to everyone in their address book.[18] You should also consider the possibility of a worm or hackers altering your "safe" site and including one of these exploits. Worms, trojans, and viruses are becoming more sophisticated and are beginning to use a mix of Web and email for their distribution.

The SANS/FBI report looks at IE as being threatened from your internal Web administrators. Many businesses have internal use only Web pages that show statistical reporting, business memos, Web-based email, and other information deemed important to the business. Web site administrators could use the aforementioned exploits on these internal pages to grab the CEO's files, email authentication cookies, or other information directly from his machine without ever having logged into the victim's machine. Disgruntled employees with personal Web sites might easily be able to social engineer someone within the company into visiting an exploit page on their site as well.

Prosecution could be potentially difficult in these situations. The fact that there is no login and, in most attacks, no software installed makes it difficult to identify that there was ever even an attack. If the victim was visiting a trusted internal Web site, then the traffic won't stick out in most logs. By the time a problem is realized, the browser cache may already be overwritten and the malicious page would be changed back to normal. People may not even think to look at these vulnerabilities as an attack mode since they are not as commonly seen.

The Web browser is not a daemon server so a direct attack cannot be launched against it. Most browser attacks are more of a trap situation, where the attacker attempts to lure the user in or plants the trap somewhere the user is sure to go. This makes it great for targeting an individual or small group. If an attacker needs to do an active attack, they could target a server on their system that talks to the browser, such as an instant messaging client, and get that software to deliver the attack.

Types of Solutions

The prevalent use of Active Scripting on the Internet does not make disabling it in your browser an easily viable solution. Luckily, there are as many protections against these problems available as there are types of attacks. Each of the solutions has a different focus and should be looked at in terms of your corporate or personal needs.

Traditional free methods: You can frequently patch your systems, but not all problems are patchable, so this should not be considered a solution in and of itself. Patching and upgrading the browser should be as equally important as updating the OS. It would also be helpful to make sure your browser security settings match your policies. I would also recommend including surfing security as a part of your company's security education program.

Corporate solutions: Employee Internet Management (EIM) are corporate Web s filters, such as WebSense and Surf Control. These help to limit your employees to viewing traditionally safe and can block active content based on file type. Corporate

18. *http://www.cnn.com/2002/TECH/ptech/10/28/security.net/index.html* – E-Mail Greeting Card Hides Porn.

content filters, such as Finjan's SurfinGate and Alladin's E-Safe offer software for your gateways to filter active content on both Web and email, which is a little more permissive then just blocking. Their design allows them to be combined with antivirus and EIM software to give well-rounded, continuous protection for your office. Finjan's SurfinGuard executes all code in a sandbox allowing you to fine-tune control of active content.

Mobile protection: Anonymous proxy systems such as a service-based Anonymizer can strip and/or filter active content that may be malicious. These have the added feature of masking any research your firm may be doing on the Web and adding encryption to prevent sniffing. Recent versions of personal firewalls such as Zone Alarm or Norton can be configured to block, warn against, or allow potentially harmful Web content. Desktop antivirus software can also pick up on some Web attacks that have been used in viruses.

Current Status

Every time that I hope they have found all the holes in IE, there are three more postings on BugTraq. In my opinion, Microsoft coupled their browser too tightly to the OS, and to software development related to the OS, to provide good security. This isn't to say other browsers are pristine. Microsoft is definitely not the only group struggling to deliver the full power of the Web securely. Mozilla, which is the basis for many browsers, has had its fair share of browser exploits as well. Mozilla-based browsers don't get as much media attention or hacker attention because of their lower usage. They report to have 9.6% of the market.[19] Even Lynx, which is a text-based Web browser for UNIX, has had security vulnerabilities.

Unfortunately, I don't believe any browser group will ever be able to deliver the complete power of the Web with complete security all on their own. The browser is expanding it's roles and responsibilities to handle more and more technologies. A study last year determined that the Linux 6.4.2 kernel had 2.4 million lines of code and that Mozilla had 2 million lines of code.[20] Considering that Mozilla is the basis for more sophisticated browsers such as Netscape, an advanced browser can exceed the size of the Linux kernel. Browsers are becoming increasingly powerful and complex.

Based on this, I believe that the number of exploits in browsers and the high degree of browser interaction with other software will continue at its current rate for some time. The expanding size and scope of the browser makes it a more and more tempting target. Viruses, trojans, and worms are increasingly using HTTP in addition to mail as their mode of delivery. It is only a matter of time before more scripts become widely available for the script kiddie community to exploit. Fortunately, most hackers who have found holes are currently content with proving the holes exist and not going further.

On the upside, every browser hole that is fixed is one less hole to be found and one more lesson learned by the browser community. The other browsers in the market need the same level of scrutiny as Microsoft. Hopefully, the other browser teams are watching what is going on with IE and reviewing their own code for the same problems.

Third-party vendors are still behind in handling the newer Web technologies. People need more of an option than just disabling active content. Although filters for scripting have come a long way, the industry is still lacking in filtering various media types.

19. *http://www.infosecuritymag.com/2002/nov/digest07.shtml#news3.*
– Six Pack of Mozilla Vulnerabilities Discovered.

20. *http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html.*
– More Than A Gigabuck: Estimating GNU/Linux Size.

RELATED LINKS

http://www.Websense.com – WebSense.

http://www.surfcontrol.com – Surf Control.

http://www.esafe.com – Alladin's E-Safe product.

http://www.finjan.com – Finjan Software.

http://www.anonymizer.com – Anonymizer.

http://www.zonelabs.com – Zone Labs (Maker of Zone Alarm).

http://www.symantec.com – Symantec.

http://www.anonymizer.com/snoop – Privacy analysis page.

The filters and sandboxes for these technologies need to be advanced and in some cases invented. It would be advantageous for the browser groups to work more closely with third-party vendors in helping to secure the content coming into the browser.

In addition, education of the general public on issues related to browser security will need to be improved. Along with touting more advanced features, browser groups should also emphasize the increases in security that come with the upgrades. There are too many technologies involved with the Web for the average user to comprehend each one, but they need to know there is protection out there and that they should be using it.

Conclusion

The important thing to remember for the corporation is that Web browsers are more than just a threat to your employees' productivity. You have to consider the threats from both outside and inside your business. Home users need to learn that their Web browser is as vulnerable as their mail clients. Due to the increasing importance of the Internet, Web browsers are evolving into mini-operating systems, and patching them should be taken as seriously as patching an OS. If an attack occurs, administrators will have trouble identifying the source of the attack since there are few clues left behind on the victim machine. Fortunately there is supplemental software and services available to protect against the large number of unpatched vulnerabilities in Internet Explorer that leave Windows exposed to an attack. It is sometimes hard to comprehend just how much of a threat Web surfing can be. After all, how much harm can a little Web page do?

will the real "sysadmin of the future" please stand up?

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

Why do I think that system administration is a growth industry, in spite of the economic climate of the last 12 to 18 months? Why do I also think that system administration will grow increasingly unrecognizable to current system administrators? Is the professionalization and specialization of the field going to become the latest planet-killer to my cohort of generalist dinosaurs?

As context for some of these questions, let's briefly recap a few points from my previous article, "Vive la Révolution: Now Get Over It."

> "Let me explain. (deep breath) No, there is too much. Let me sum up."
> Inigo Montoya, *The Princess Bride*

Our technological revolution was both eclipsed by and conflated with the economic boom cycle of the past decade. The false lesson that the business world learned from the dot-com bubble burst was "OK, we were right all along; time to go back to sleep." The general public promptly lost faith in the real power of technology for social change once their 401Ks started plummeting. To most laypersons, the Internet is now snake oil.

The dot-com bubble was information technology's Watergate. In the long term, more and more people will realize that the worst damage of the burst was done to our cultural perception of technology. Over the next decade, that will vastly eclipse the drop in fiscal valuations. My best friend and I were born only five years apart, but he can still remember when everybody trusted the government. I, who grew up with Watergate in grade school, can't even remotely imagine that. Some of you industry veterans reading this can't imagine that anyone ever believed in "the New Economy." Others, who came to this field within the past decade, may still be cursing with frustration. I know many people who believe that if the Suits and VCs hadn't panicked we'd all still be employed – and, possibly, our society as a whole would be headed in a healthier direction. To you in particular, the past six or seven years look normal and the last 18 months look like an insane overreaction.

At the same time, technology has become a startlingly transparent conduit for content delivery. Business data, transactions, and entertainment are traveling through a wide variety of networks, devices, and media and being accessed in a less specialized context by a wider variety of users. Most people give no thought to the vast array of infrastructure that provides 120 volts AC and telephone dial tone in their house. Network and content are already being treated similarly. They are relied upon without being understood. They are taken completely for granted with no perception of the operational difficulties and logistics involved in keeping these services available.

Before Enlightenment, Carry Water, Chop Wood

This change in long-term mind-set has largely bypassed the sysadmin community. Ever pragmatic, we realize that fewer dot-com Web farms doesn't mean the end of sys-

Dead ends are viewed from the forward-moving perspective of success and rarely recognized while one is traveling down them!

tem administration. The usual tangles still need unsnarling, whether you are sitting in Aerons at a VC-provided incubator office or parked on lab benches somewhere in a bunker. We also got to see a lot of stupid business decisions from the trenches, and by and large didn't give up our faith in technological advances just because we were almost constantly ordered to misapply technology by people who didn't understand it. And, of course, "looking under the hood" is what we do for a living. The better the packaging, the more suspicious we get that there's nothing worthwhile inside the package.

An ugly side effect of the transparent technology conduit is that Joe and Jane Public's reactions to technology are now *almost entirely* based on the packaging. Does it look shiny, smooth, rounded, high-resolution color, like TV, cable, cell phones, pagers, iBooks, and graphic-heavy Web pages? If so, it's a common everyday object and should "just work." Does it look angular, knobby, industrial, option-heavy, bristling with cables and interfaces? Then it's "high-tech" and probably a boondoggle – get rid of it!

Unfortunately, many infrastructure technologies and service building blocks fall into this category. Even worse, Joe and Jane Public, in their capacity as middle managers, department heads, and CFOs, are making technology infrastructure decisions informed largely by consumer-technology-marketplace conditioning, filtered through their reaction to the dot-com bubble. Ouch.

The increasing transparency of content helps to feed the growing perception that sysadmins are not necessary for everyday business and home computing services. Even the non-techie early adopters, traditionally allies of the IT department, have picked up the consumer-appliance view of sophisticated technical devices. Their smart MMS phones, home media centers that double as computers, and voice-driven GPS navigation aids seem quite routine. Their experience of the ease of use misinforms their view of the ease of deployment and leads them to devalue the role of systems architecture.

The evolutionary pressures on technology are increasing. Many people define evolution as "progress," but those of us with some sciences background may recall evolution's definition of progress: something that works until it doesn't anymore. The dinosaurs were the pinnacle of evolution until that darn comet came along. Mammals were merely experimental window systems implemented in old LISP dialects. Dead ends are viewed from the forward-moving perspective of success and rarely recognized while one is traveling down them! Circumstances and changes determine what will prove to be a bad path. The consumer marketplace is performing technology selection based on what is selling right now and what will take the least time to get to market. Meanwhile, we're heading down a rat hole where technology is supposed to manage itself (it isn't) and communicate in a plug-and-play fashion (it doesn't) to meet our larger cultural and societal needs (as defined by whom?). The role of system administrators as knowledge workers rather than janitors is questionable in the market space that is evolving.

The good news is that it's not yet too late to change this. The bad news is that the situation is not going to solve itself if we sit and do nothing. Our opportunity to add actual value to the technology process, instead of babysitting Web farms, is unparalleled at this time. The catch is that we are going to need to work hard to educate the people who need us most, the consumer technology industry. These are the folks whose output to the GNP dwarfs that of the dot-coms and the whole computer industry even at the stratospheric height of the bubble. They are also the folks who make, in

addition to our home espresso machines and 12-volt automobile mini-fridges, many things we think of as cool toys. Hey! This doesn't sound terrible yet.

A number of folks may counter that it's hard to stay optimistic when you're looking at unemployment benefits running out. I agree wholeheartedly. The recent plummet in the job market for system administrators is reflective of more than the current economy. It's also where the rubber meets the road for "sysadmin awareness." Many businesses out there might have made different choices about who to lay off if they had a better understanding of what their systems staff was doing for them. A business may decide to let things slide in a time of recession, but they rarely choose to let the infrastructure go completely to heck in a handbasket. Some firms will soon find that is exactly what they've chosen. Others are discovering it right now.

> "The future is fun. The future is fair. You may have already won. You may already be there." – Firesign Theatre, *I Think We're All Bozos on This Bus*

Let's take an example of new consumer technology: a vacuuming robot for household use. There's a small company making one of these, but at some point it will be eaten by a larger company or subdivision of a multinational. Those are the people who will be hiring you or me if we don't like to work for small edgy companies. Those are the people who have the money to fund product development. The great thing about product development is that it is a revenue expense, not an overhead expense. It is the most neglected place for sysadmins on the *friendly* side of the balance sheet.[1] Changing this will not be easy, but it is very possible and will be very worthwhile.

Our vacuuming robot is real, and it is called "Roomba." At one time these were vastly expensive research toys. Enter the $200 version, available now from various high-end "tech-toy" stores. Roomba can navigate within individual rooms of a standard household environment and is designed to require minimal intervention. Among its bag of tricks are several types of space-traversing algorithms and a receiver for an "Invisible Fence" broadcaster, which tells it where it isn't welcome (or could fall down stairs). Finally, something for all of us who have watched a swimming-pool cleaning robot at work and wished we had one that would do the floors at home.

As with any newly deployed technology, the Roomba is far from perfect. A review praised its abilities and addressed a few minor shortcomings: "Roomba shuts itself down when an object gets wrapped around its main roller, but it leaves it to you to guess what happened. I would like a better battery indicator. And in a few years time, I would like a machine that can automatically wake up when I am out of the house, clean the floors, and then plug itself in for a recharge." These are fundamentally consumer issues, but let's look at how we might address them from a system administration perspective.

The Invisible Fence signal could be modulated to carry data, saying "keep out of this area" or "follow me to your charger." A tiny wireless transmitter can send standard notifications such as "main roller stuck" or "ready for next room." Let's plan for multiple Roombas from the very beginning. Multiple Roombas are more likely than an affordable "ÜberRoomba" capable of mapping the whole house and handling obstacles like stairs. Thus each unit should come with its own RFID serial number and/or MAC address, so that it may communicate uniquely with the house controller.

House controller? Indeed – small house or apartment local area networks are becoming a standard feature in high-end "smart houses." It's easy to build small repeater-style

1. Those of you reporting to engineering groups know what I mean by "the friendly side of the balance sheet." So do those of you supporting production systems, unless you have a particularly clueless manager. Being perceived as adding to revenue is the single best defense anyone has against layoff or marginalization. Start reading Tom Limoncelli's soft skills LISA papers right now if you don't grok this. The job you save may be your own – or your next one, if you are looking right now.

2. Interestingly enough, the executive team that acquired Palm for US Robotics planned to make its major revenue off the cradle, not the device. A recent article quoted one of the team as saying that "although there might be one or two handhelds in the home, consumers would have as many as a dozen or more cradles."

monitoring stations, plugging into a handy AC outlet. These stations could listen for alerts and roll them forward into a central loghost. Alerts would be acted on according to the owners' configuration, triggering outcall paging or emailing, or changes to a wiki or Web log. For that matter, each room of the house could have an inexpensive aggregator that handled traffic for that room. The "big red button" for a room could be configured to have various meanings – "route my calls here," "turn all my appliances on," or even "call a Roomba, I just spilled my corn chips!" A true two-way network would be scarcely more expensive to deploy, especially if the first kludge, excuse me, implementation was merely to do two-way to room nodes only. The big red button by your exit doors could signal Roomba, and all your other house appliances, that you were heading out for a while, so now would be a good time to vacuum, run the dishwasher, and so on. Security mavens will quickly realize that it's also critical to encrypt or broadcast limit so that one doesn't also signal another monitor station that someone may have stuck under your porch rail.

If implemented poorly, RFID or wireless beacons on advanced consumer tech-toys could be misused in scenarios ranging from wireless peeking into wrapped birthday or holiday presents to easy drive-by house casing for neighborhood burglars. What are the chances that the lessons of decades of IT deployment and computer security will be forgotten in the rush to market? A typical example is the inept deployment of wireless cash registers at certain consumer electronics retail stores – customer credit card data was wirelessly sniffable from the parking lot. Help! Where's a systems architect when you need one?!

After Enlightenment, Chop Water, Carry Wood

> "The purpose of IT is to seamlessly and transparently provide the other nine-tenths of the iceberg for people who need to work with chunks of floating ice."
> – Strata R. Chalup

> Weinberg's Second Law: "If builders built buildings the way programmers wrote programs, the first woodpecker to come along would destroy civilization."
> – Gerald Weinberg

Technologies like clockless computing may soon revolutionize miniaturization and price points of ever more powerful systems. Clip-on GPS transponders for your child's backpack, or your girlfriend's cell phone, are becoming available. GPS technology has become cheap and entertaining enough that art/sociology projects like Amsterdam Realtime (a map of Amsterdam constructed entirely by the actual movements of people equipped with tracer units) are becoming easy to implement. The personal robotic vacuum cleaner is here. And yet – have we really invented anything new? Think about what we spend most of our time doing as system administrators, namely systems integration. Where are the systems which are designed from the ground up to be integrated?

As I look at the entire history of computing, PDAs are the only things I see that are *really* new. The Palm Pilot, and the rest of the PDA market which it created, is new and unique because PDAs are the first computing devices designed from the ground up to be used *by* an individual *for* the purpose of synchronizing and integrating into an environment. Expand the term "PC" and you get *personal* computer. A computer of my very own, which I don't have to share with the mainframe users. I'll process my own little batch jobs by myself in my own little world. In mainframe land, systems like virtual images created the illusion of a computer of one's own. This was seen as a nec-

essary interface to the shared computer. Inherent in both the mainframe and PC world is the idea that computing zones are little fiefdoms, separate and inviolate. They're simply not designed to be part of a greater whole. That little PDA with your calendar, address book, and online newspapers represents a true paradigm shift.[2]

We may be on the verge of losing this insight. PDAs running Windows CE and Linux may be cool and fun, but they're essentially portable PCs. Their quality of integration, of being a view into a larger shared data set, becomes merely another function. A PC in a PDA package is still a PC, as indicated by the increasing use of the term "handheld," for *handheld personal* computer. It's not meant to be integrated, and that's a built-in limitation. That limitation is being inherited as an unexamined assumption by new devices, such as the Roomba in our earlier example. The list of hypothetical improvements to the Roomba are all about functionality within an integrated house-wide system.

This prefigures the "smart house" as its own little neighborhood of smart devices. These devices are inherently isolated from one another, speaking only to "the controller." In our example, we held to that convention so as not to introduce too many things at once. Think of how much more functionality we could have by making each of the smart devices as quasi-autonomous entities which are preconfigured to know how to "behave" in the context of a greater whole. For that matter, none of the smart houses that I have seen documented, even Microsoft's widely touted vision, make the next step into a "smart neighborhood." People are hungry for the benefits of compatible systems and shared data, as evinced by the recent peer-to-peer software explosion. Yet few of the first P2P systems provided ways to limit the amount of data one shared, or defaults other than "my whole hard disk."

It doesn't take a great leap to map many of our security, data privacy, and systems maintainability problems onto our assumption of "personal" computing. Many of these problems can arise directly from a mental model of traditional boundaries. This mental model fails to recognize that characterizing data as a physical object, "hidden" files as actually hidden, and so on, does not reflect the real transparency and permeability of information. The necessity of shaping memes of privacy, security, and maintainability ought to be revolutionizing the field of system administration. Instead, we spend all our energy on reactive strategies that by definition will never solve the problem. Of course, bigger and bigger problems are on their way – wireless networks and broadband will not create new problems so much as provide a rich agar to nourish the swift spread of existing problems.

Here is one of the many opportunities for active, not reactive, system administration. Again, a vast market is poised and waiting to leverage what you build. You can't be the CEO of a killer company and retire at 35 on the profits. It's not that kind of market space. What you can be is an incredibly respected and valued employee or consultant, having a blast making reality out of things that were always glossed over in science fiction. You can also, if you wish, help build the "moral high ground" in data privacy and accountability. New tools and standards, which you can help create, change the way that people think about privacy and information. I'm eager to see free software written to build customer expectations of "good" UI and information control, before the commercial stuff really hits. I estimate there's a gap right now of at least 12 to 18 months where a Bluetooth or 802.11 "universal remote" or "neighborhood integrator" type of program could find a niche. The right tool/program could embed itself so deeply into the user community that they'd never be satisfied with a non-integrated solution,

Wireless networks and broadband will not create new problems so much as provide a rich agar to nourish the swift spread of existing problems.

much less one that doesn't preserve privacy and configurability. Look at Napster as both a good example of creating an expectation space and a bad example of how to handle privacy and security issues.

Numerous studies have shown that the penetration of PCs and handhelds has never approached that of television, VCR, CD, DVD, and the like. The determining factor is always listed as "ease of use" but might instead be phrased as "what can I do easily." All of them use tools, programs, and objects that have more features than we really use. Unless one has a strong need for a complex, hard-to-configure feature, one just uses the easy features. One of the answers to the perennial question, "What do sysadmins really do?" is my smart-alecky iceberg quote above.[3]

A clearer, more direct version of that answer is that sysadmins serve as an integration buffer to make things transparent to users. There is now an enormous industry based on enabling ever more sophisticated transactions and capabilities to be harnessed by so-called "naïve users." This industry may not yet understand that it needs us. We should be out there demonstrating beyond doubt that our expertise has value in this domain. How do we get there? One path involves taking the next steps to professionalize system administration and to dramatically increase its academic rigor.

> "The other day I heard a person in storage systems dismiss the efforts in quality of service research being done in networking as just people who look at packet loss. By that flip remark, he dismissed many good ideas which would have advanced his research. . . . It is easier to try to re-invent than to read past literature and filter the good from the not so good . . . . I know there are exceptions to what I have ranted about but from my experience they are few and getting fewer. It is time for the senior people in the field to demand that people behave like scientists."
> – Dave Farber, CS/IT/Telecom *éminence grise*

Here's another example of where we need to push back, as a profession, against the perception that our skill set is limited to taking care of what others have built. We're seeing the proliferation of unplanned, emergent systems used for important business and consumer activities. These systems are being built with an eye toward accomplishing market goals and minimizing capital expense. They are often assembled out of ad-hoc components or networks from failed competitors and are minimally integrated. They are not being designed for security, or maintainability, or stability in overload conditions. The most egregious visible abuses are seen in the area of wireless, since those make excellent press right now. As we know from experience, there is no area or function that somebody can't try to put together with bubble gum and baling wire. Even worse, secondary systems are being planned and implemented which take these jury-rigged systems as a given.

These systems are constructed primarily of software and applications rather than hardware, yet Joe and Jane Manager tend to regard them as "the network" and thus out of the domain of traditional system administrators. In fact, most of their value lies in doing more or less traditional Internet or intranet transactions at the edge of the protocol network, or in emulating traditionally sysadmin'd TCP/IP services across a wireless network. Our profession understands more about the inherent instability and trade-offs involved in making these services work than do traditional telecom engineers, yet we are increasingly out-of-the-loop in design and deployment of these next-gen services. It's possible, even likely, that some of these services will be composed in a way to preclude some of the more irksome fundamentals. One example is prototype

3. Thanks go to Benjy Feen for correcting my initial use of "glacier" in the phrase!

file-sharing networks that store multiple copies of a document, handling locking, versioning, and references without explicit user intervention. Wow – no more backups! There's a little Catch-22 here, which I'm sure you've already spotted. Without knowing the fundamental concepts, such as backups, the implementers would not have known the desirability of such a feature.

This is all a train wreck waiting to happen, and/or happening now. Sysadmins will be needed to "save the world," but first we have to convince the world. Even a few train wrecks won't make the world beat a path to our door unless they perceive us as *much* more highly trained and capable than they do now. We need to demonstrate not only our technical specialties, but formal problem-solving skills and knowledge of specific, already recognized domains of engineering.

With opportunity comes responsibility. We must integrate with traditional professions and take on more academic rigor in order to maximize our contributions. Yet we must also retain aspects of an independent specialty to retain a voice outside of other professional domains. We must continue to professionalize ourselves and to aggregate a specialized body of knowledge and group of best practices, while at the same time reaching out to the engineering disciplines. Cultivating a "guild" mentality will only ensure that conventional engineering specialties reinvent our wheels and acquire systems experience within the tenets of their established domains of knowledge. Worse, they will make all of our old mistakes, just as they are now, but with real-world systems like automated subway cars and wireless cash registers. A look at the RISKS Digest archives should be enough to convince even the most hardened skeptic.

Me? A Sysadmin? No, I'm a . . .

As the profession matures, we will find ourselves less isolated and more integrated into the "normal" flow of professions. I think that we will find this mainstreaming comes with something of a price: System administrators in large part will lose their identity as a specific profession. Instead, we will come to view system administration as a body of knowledge and a collection of skill sets. This has already happened, from the other direction: There are sysadmins out there who don't know they are sysadmins or who view administration as one of their technical job responsibilities.

A couple of years ago, changing planes for my flight to LISA in New Orleans, I met someone in the airport who really made me stop and think. He was carrying a book, whose title escapes me right now, that clearly identified him to me as a system administrator. I struck up a conversation with him, thinking he might also be on his way to LISA.

He had never heard of LISA or USENIX. He was familiar with SAGE and was surprised to hear they were sponsoring a conference of which he was unaware. After all, as a manufacturer of cutting-edge A-to-D chipsets, SAGE was well-represented on the shop floor, and his group had purchased a number of their offerings. His full-time job was to run a production optics shop making lenses out of specialty materials for manufacturing use. His computer-controlled lathes and production equipment were run by Solaris and Linux boxes. He had a degree in materials science and thought of himself as an optics technician. I showed him the current conference program and talked to him about any "open issues" he might be having with his machines. He was completely convinced that things like (our) SAGE, USENIX, and the LISA conference were of no interest to him. His *nix boxes were simply front ends for his tools. Sure, he could patch the OS, do new installs, etc., but that wasn't his *real* job.

REFERENCES

"Vive la Révolution: Now Get Over It!"
http://www.usenix.org/publications/login/2002-10/pdfs/chalup.pdf

http://www.virtual.net/Ref/pubs/Dec02-Vive-la-Revolution.pdf

Round 2.0 ("Just Because the Internet Hasn't Yet...Doesn't Mean It Won't")
http://www.contextmag.com/archives/200208/Insight2Round20.asp?process=print

Controller Pushes Ethernet into Everyday Equipment
http://www.eetimes.com/printableArticle?doc_id=OEG20021122S0031

Firesign Theatre
http://www.firesigntheatre.com/

iRobot Roomba
http://www.technologyreview.com/articles/print_version/wo_garfinkel100902.asp

http://www.roombavac.com/

Wireless Cash Register Micro-Debacle
http://zdnet.com.com/2100-1105-898775.html

http://www.computerworld.com/mobiletopics/mobile/story/0,10801,72024,00.html

Dave Farber, "A rant ... on the state of our field in research"
http://lists.elistx.com/archives/interesting-people/200210/msg00073.html

Clockless Computing
http://www.nytimes.com/2001/03/05/technology/05IVAN.html?pagewanted=print

http://www.technologyreview.com/view/article.asp?p=11649

(abstract at *http://www.technologyreview.com/articles/print_version/tristram1001.asp*)

Tollboth Technology Goes Mainstream
http://www.nytimes.com/2002/07/07/business/yourmoney/07PAYY.html

RFID Fun for Everyone
http://www.rfidjournal.com/news/nov02/gillette111502.html

http://www.rfidjournal.com/news/july02/alien7102.html

http://slashdot.org/article.pl?sid=02/11/17/0327244&mode=thread&tid=126

Peace of Mind, or Minding Others' Business? POMALS
http://www.wired.com/news/business/0,1367,55731,00.html

Amsterdam Realtime Project
http://www.waag.org/realtime/

The Risks Digest: Forum on Risks to the Public in Computers and Related Systems (comp.risks)
http://catless.ncl.ac.uk/Risks/

Wood, Water, Enlightenment
http://www.donmeh-west.com/enlighten.shtml

http://www.amazon.com/exec/obidos/ASIN/0874772095/virtualnet-20/

A Dystopian Vision, and Some (Widely Varying) Smart Houses
http://www.c4vct.com/kym/humor/shouse.htm

http://www.cc.gatech.edu/fce/seminar/fa98-info/smart_homes.html

http://www.infocontrol.com/examples01.htm

http://learnat.sait.ab.ca/ict/cmph200/smarthouse.htm

http://filebox.vt.edu/users/mikemike/smart-house/

http://www.bath.ac.uk/bime/projects/smart/

http://future.newsday.com/3/fsmart7.htm

http://www.eren.doe.gov/consumerinfo/refbriefs/ad7.html

http://houseandhome.msn.com/Improve/MicrosoftHome0.aspx

http://www.usatoday.com/life/cyber/tech/review/crg976.htm

Murphy's Law Origins, and Others
http://www.lylemariam.com/murphy.htm

Lakoff & Johnson, Metaphors We Live By
http://endeavor.med.nyu.edu/lit-med/lit-med-db/webdocs/webdescrips/lakoff1064-des-.html

http://www.amazon.com/exec/obidos/ASIN/0226468011/virtualnet-20/

http://www.uoregon.edu/~uophil/faculty/mjohnson/mjohnson.html

http://www.linguistics.berkeley.edu/lingdept/Current/people/facpages/lakoffg.html

Looking at the job marketplace for engineers, project/product managers, and quality assurance people, one finds job ads aimed at various "flavors" rather than a generic "engineer" or "project manager." Common orientations are civil engineering, aero/astro/military, biomedical, pharmaceutical, manufacturing. A civil engineer is not the same as a software engineer. A pharma/biomed project manager is not the same as an IT project manager. As system administration becomes more normalized and professionalized, individual sysadmins will find it increasingly necessary to have expertise in some conventional domain of engineering sciences, or commerce. Acquiring the expertise on the job, as we have done in the past, may not be a valid option in the future. System administration skills will be seen as a necessary but secondary component to the domain knowledge, and good schools will offer system administration electives which engineers and scientists can use to equip themselves to do much of what we do.

Sysadmin: The Next Generation

We are at a crossroads, where individual careers can take many paths, but the profession as a whole needs to move in certain directions. System administration needs a well-defined body of knowledge, formal course curricula, careful attention to research & publication, meaningful certification programs, and increased cohesiveness and commitment to evolving as a profession.

As more sophisticated technology is deployed in day-to-day living, the lessons we've learned can help prevent negative outcomes along the continuum from disappointment to disaster. Civil engineering and finance are just two of the areas where good outcomes are mandatory, and both the public and private sector will spend money to ensure this.

In order to enable the transparently functional technology demanded by the consumer-technology market, we must become obtrusive and insist that a systems management perspective be applied. Since consumer goods constitute one of the largest pieces of the economic pie, jobs are now and will be available as our value is recognized.

The individual paths look very odd and nontraditional to those of us who have been here a couple of decades. Some may decide to stay the course as generalists and end-use integrators rather than moving into the technology-creation process. Others are looking for the next niche to move into, or the next migration path to follow. All of the directions indicated in the references are navigable for those who are just starting out as well as those who are wondering where to go next. All of them represent well-funded but underutilized employment demographics with opportunities along the whole spectrum from struggling startups to established companies. Self-promotion, re-training, on-the-job learning, and a certain amount of "street smarts" are likely to be necessary to make the jump into some of these parallel tracks. They may be options you never realized were available. Not only are they out there, but we need to cultivate them to ensure the future of system administration as a profession.

confessions of a sysadmin turned salesman

by Steven M. Tylock

Steve Tylock has been managing infrastructures for the past 15 years in the Western New York area, and helped organize GVSAGE as a local SAGE for the Genesee Valley region.

Stylock@gvsage.com

I have no regrets.

I still consider myself a system administrator. I'm not sure why though…

It's been quite a while since backups and restores were my responsibility. Haven't added a user since I don't know when. The last time a system crashed, I was the one who got the call letting me know how long it would be down.

But still – I can be a sysadmin, can't I?

How Did I Get Here?

I was definitely a sysadmin at the university, that much is clear. The professor offered me the job that May I graduated. Sure, it was hourly until his grant came in, but the grant did, and I got the position (even though it had to be posted). My title was "Computer Programmer," but they didn't have anything closer. All the elements of a beginning sysadmin were there – a more learned mentor or two, figuring out that automation is a good thing, first exposure to USENIX (no SAGE then, sysadmin topics were part of the main conference). By the time I was leaving I had ported a driver, broken in a couple of undergraduate assistants, and submitted my first paper to USENIX.

Most of my tenure in industry was as a sysadmin. Sure, I opted for the switch from the school setting to business – better pay, moving to a city I wanted to be in – didn't everybody, eventually? The title was "Software Engineer," but several years after I'd left they adopted a series of "System Administrator" titles. The first few years flew by as an "individual contributor": creating a program to mentor admins and techs, pulling some of the related departments together, upgrading OSes to consistent and current levels. There was that stint as a software developer for a product, but it lasted only nine months – the project was successful, but the work didn't agree with me.

Maybe that jump back into sysadmin was the start of it – creating a "self managed team" with the three of us. There was no doubt that I was the leader of the team, but it worked well. (People I bump into still think of it as the best environment they ever had.) Maybe it worked too well. After a few years there weren't as many complex issues, and management didn't show me a path forward.

I learned a lesson about titles with the next switch. "Systems Manager" sure looked like a management position on the internal posting . . . too bad it meant manager of systems to some of the people. It had elements of hands-on system administration though! Who knew I would have to be the one closing the MRP system at the end of each month. And even after I added the two technicians and the three engineers, I was still in the thick of it helping and mentoring them.

Jumping to a consulting firm didn't change what I was doing much. They called me a "Senior Systems Architect," but it was just like system administration except that I wasn't overhead anymore, and the client actually wanted me there. It was definitely a step back toward technical administration. Too bad it degenerated more into staff augmentation than real consulting. But that pushed me to the director position, so it can't be bad.

I thought that the one experience I was missing was being a part of something that failed.

"Director of Desktop and Network Services" – man, I liked that title! I upgraded the company's entire email structure as an audition for the position – that was sysadmin work. Of course, after getting the position, my direct work was cut way down. But that was best for the company and the team – they needed help getting the individuals to work together. Politics, budgets, and an infrastructure that was being held together on the backs of overworked, underpaid and under-appreciated tech staff. What bigger challenge could I ask for?

While there were times I'm certain the guys thought I was unfit for actual sysadmin work, I think they ultimately realized what I was doing and felt better off for it. True, I spent a great deal of time straightening out the budget and dealing with management, but I was there when they needed me – helping guide them, breaking up the Windows-UNIX fights, pulling together short- and long-term plans for the infrastructure. We built an impressive list of accomplishments in a fairly short time.

I started looking before the end came, and we were still doing fairly well then, but oddly enough I thought that the one experience I was missing was being a part of something that failed.

True, the company still has its doors open, but it is not the same place that it was. They dropped consulting and project-based work in favor of selling software. While they needed a reasonably sized group to manage the five sites and 200 employees with continual demand for special projects, it doesn't take too much to keep 40 people and two sites working.

Close but No Cigar

I was out, pounding the pavement, but that was okay.

While it was not as nice for my bank account to be on the outside, it was a whole lot better for my head. The family and I struggled through by doing what had to be done.

I've always thought it would be no problem finding a job if I really needed to. Who would know that the conditions would get that tight for so long. While there were occasional situations that looked good, simple odds were not in my favor. At this point, I wasn't an easy fit – too experienced for lots of situations, and not experienced enough in management for those infrequent positions.

While I didn't have a job or a title, I sure felt like an out-of-work sysadmin.

Going a Courting . . .

The beginning of the end came at a technology show (see the February 2002 issue of *;login:* for more info on this topic). When I wasn't staffing the GVSAGE booth, I was walking the floor and networking.

One of my walks took me past the booth of a salesman I knew. We had brought their equipment in for evaluation, but had been unable to convince management that we should invest (in hindsight, the reasoning seems *sooo* obvious now). He said we should talk more.

The show was good to me; I had a number of leads to follow up on, one being with this VAR. Pre- and post-sales work – I could do that; it's like showing the customer how the product fits in and then making those connections.

But . . .

The shoe fell when I found out there would be pre- and post-sales work available when I found prospects ready to buy. Sysadmins are generally smart enough to put two and two together, and I'm no exception to the rule. This was a position in sales.

Sales . . .

So I did what any engineering-minded person would do: I began studying the sales profession. I "did lunch" with a couple of sales people I liked and respected. I started reading everything I could get out of the library on the topic. And I kept working the other job leads I had going.

A Whole Lot of Emotions

The world of sales is a bit different. The tie from "success" to "reward" is pretty darned direct. You are not contributing to a project, or providing the environment for the project to succeed in, you are helping someone else get to the point where they think that the product is worth purchasing. The sale starts the flow of money into the company – without it, the company stops existing. If the product costs the company $X, and the customer is paying 150% of $X, your company is "ahead" by that 50% of $X.

At the most basic of levels, you are paying yourself out of that percentage. The upside is that exceeding expectations should be both immediately obvious and rewarding. The downside is that lack of success is also readily evident with notable consequences.

The exact distribution of this success varies by sales domains and organizations. Safety for the sales person comes in the form of a base salary. Risk and reward comes in the form of commissions.

So while I was excited about the prospects ahead, the fear of failure had financial as well as mental strings attached.

Don't Give Up the Day Job

The first iteration on "I'll do it" was – what if I found another part-time contract sysadmin job that would allow me flexibility. That's what wannabe actors do, right?

As luck would have it, I kept moving forward on all of the potential job fronts, but every other opportunity dropped out – until I was left with just the sales position.

And somewhere in there it hit me – why am I resistant to this position? If something like this opened up at one of the big manufacturers, wouldn't I be banging on the door asking for it? I knew it was something I could do, but did I want to? (It's not like I'd be selling snake oil; the product was one that I had looked into buying myself!-)

Just Because . . .

Just because some other sales people push bad products on unsuspecting customers doesn't mean that being in sales is bad. And that image is what I didn't want to become.

On the other hand, I've seen people in the USENIX / SAGE world move into sales positions (when you take a piece of software and make a company devoted to selling it or services related to it, you are in sales;-). I've seen them get out of sales positions, and that downside isn't as bad as I'd imagined either.

Just because some other sales people push bad products on unsuspecting customers doesn't mean that being in sales is bad.

Signing on the Dotted Line

So I was convinced, and the deal was worked out. I like to think that we negotiated to the point that encouraged me to act "hungry and motivated, but not desperate." The VAR wants to motivate sales, but not at the expense of longer-term relationships with the customers. I needed to make sure the bills could get paid in the short term, and the potential was worth my effort.

Reactions from my family cover the gamut. Some are positively enthusiastic, some are unsure but supportive, and one thinks I've made a mistake. True to form, I'm out to prove the naysayer wrong.

But I Can Still Be a Sysadmin, Can't I?

As you can see, I took each of these steps in a logical fashion, one after the other.

Maybe it's like in the medical profession when a doctor joins a pharmaceutical company and no longer sees patients – they're still a doctor. Maybe being a sysadmin is not just the knowledge you have, or the activities you perform, or the title on your door. If we are a profession, and I say we are, it's got to be something more than managing user accounts.

So, yes, I am a sysadmin and a card-carrying member of SAGE, the professional organization of system administrators. (Okay, I really don't carry my card around – but I could.) And, currently, I am simply providing better data storage solutions to businesses that need them.

As I said, I have no regrets.

ISPadmin

Service Provider Book Reviews

In this installment of ISPadmin, I depart from my usual coverage of service provider technical topics. Instead, I will review several books that are of special interest to the readers working in the service provider business and, as it turns out, to most *;login:* readers as well.

In this era of moral misguidance, I feel obliged to say that I have done some work for Addison Wesley (book proposal) for which I received a small honorarium. In the same full disclosure vein, I would also like to point out that I personally paid for the three books reviewed in this column.

The Practice of System and Network Administration

If you are a practicing system administrator (SA), then you need this book. I read this book cover-to-cover, and in my 12-year technical career I can't think of another IT-related book where I even attempted to do so. Though it can easily be used as a reference when needed, it is not a technical reference book per se – rather, it attempts to document the best practices and approaches for solving SA problems. Approaches outlined and examples given are a little skewed toward larger sites, but the book still contains a mountain of information and is extremely useful for an administrator working at a smaller site as well.

A typical chapter (there are 31) contains a relatively short introduction to the topic; a section called "The Basics," the core of the chapter; a (usually smaller) section called "The Icing" covering the extras, which are less important or which may not apply to all readers; followed by a conclusion and exercises.

The conclusion and exercises I don't find to be particularly useful. However, I am not studying the material as part of an academic setting so these sections could be useful for others.

Part I, "The Principles," covers the basics of the system administration process, the essence of what SAs do every day. Material covered includes managing desktops and servers, services, debugging, namespaces, security, disaster recovery, and ethics. I found the chapters on servers and services to be particularly interesting.

Part II, "The Process," provides an excellent treatment of the various methods used by SAs to manage their infrastructure, including coverage of change management, hardware upgrades, routine maintenance, converting services, and centralization/decentralization of services. My favorite chapter in this part was "Change Management and Revision Control."

Part III, "The Practices," is a catchall, covering a number of topics not fully covered in the first two parts. These areas include help desks, customer care, data centers, networks, email, print, backup/restore, remote access, software depot (essentially NFS server housing common binaries), and service monitoring. I found the chapters on help desks and customer care to be the most useful part of the book.

Part IV, "Management," covers dealing with (and becoming) management. I can hear the collective groan, but this is an extremely important and overlooked topic. Areas covered here include organizational structures, perception, happiness, and hiring and firing. The essential chapters here are "Perception and Visibility" and "Being Happy." I have not seen these topics (among many others in the book) covered anywhere else.

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

THE PRACTICE OF SYSTEM AND NETWORK ADMINISTRATION
THOMAS A. LIMONCELLI AND CHRISTINE HOGAN
Boston, MA: Addison-Wesley, 2001. Pp. 776. ISBN 0201702711.

DESIGNING ISP ARCHITECTURES
JOHN V. NGUYEN
Palo Alto, CA: Sun Microsystems, 2002. Pp. 360. ISBN 0130454966.

The case studies, examples, and figures do an excellent job of complementing the text. Appendix A, "The Many Roles of a System Administrator" is a great addition to the book. It lists the various types of people within the umbrella term system administrator. After reading this appendix, I have a much better understanding of what drives my coworkers, not to mention myself! Appendix B, "What to Do When..." is an excellent road map on handling certain situations, like starting a site from scratch, moving a data center, etc. This chapter acts as a meta-index, tying together everything from the book, with some new material there as well.

My nits on the book are few and relatively insignificant. I think "The Extras" would have been a more appropriate section title than "The Icing." I found the 10-page bibliography (two lines per reference, single spaced) to be a little too much. It is hard to find something with that many references. Also, the length of the book (774 pages) makes it a little overwhelming to read (not to mention to revise for future editions!). Perhaps a two-volume set would have been better.

If you are a beginning or intermediate SA, you'll want this book to find out how to do everything right the first time, without learning the many wrong ways to do a task/project. If you are an experienced SA, you'll want this book to figure out why you have difficulties with certain projects or tasks time and time again. While you would be hard-pressed to get experienced SAs to agree on a single approach to anything, I would agree with 80% of the methods and advice provided in this book. It is an outstanding treatment of a topic long neglected. Every person who manages two or more machines needs this book!

Designing ISP Architectures

"This book is a model for designing architectures for ISPs of any size" is the first line of the back cover of this book. I feel that the scope is a bit larger than simply ISPs, but I will cover that later in this review. The book, part of the "Sun Blueprints" series, has a major bias toward Sun products. For the most part this is fine, but there are several instances where this is a problem. The text covers building an ISP architecture from the ground up, using an imaginary ISP, FijiNet, as a basis for the design and implementation. It starts at requirements analysis, moving through architectural models to creating a physical design, selecting components, and implementing a solution.

The first two chapters do an acceptable job of introducing the topic and deriving a design basis for the system. Some details are buried (such as what services the ISP will offer) and take some digging, but the information is there. It is unclear to me if dialup services are supposed to be offered at FijiNet, as it is not expressly stated. However, broadband services are expressly not part of the service offering.

Chapter 3 goes on to define the architecture for the service, but I can't figure out why DHCP is covered; in a dialup service provider, RADIUS would be used. I have only seen DHCP used in broadband applications (such as cable modem service), and even there its use is limited. For example, most DSL implementations utilize RADIUS not DHCP for authenticating subscribers. Another criticism would be the lack of coverage of maintenance requirements: for example, utilizing some sort of mass-update mechanism (rsynch, rdist, cfengine, etc.) in a provider scaling to 100,000 subscribers is essential!

Chapter 4 covers creating a logical design for the ISP. Once again, the usefulness of some of the figures is questionable, but overall coverage of the material here is acceptable.

Chapter 5 continues on to create a physical design. The planning capacity section is where things become very interesting. While I haven't done a formal survey, I have seen few books giving specific capacity planning formulas for sizing systems and applications. Yes, some books cover an aspect of it (e.g., Adrian Cockroft's very capable *Sun Performance Tuning*) but never from the application point of view. The formulas in this chapter are the reason to have this book. Of course, I have not had an opportunity to actually field-test the formulas, but they are a great start and would be useful for non-service providers who run ISP applications as well.

Chapter 6 covers the selection process used for hardware and software for the imaginary ISP, FijiNet. The tables in this section are not terribly useful, for no other reason than they don't attempt to be complete. Covering "Application Servers" and "Database Servers" is not very useful. According to FijiNet's plan, application servers will be a sideline business (not to mention that there are thousands of such applications). And the database server software selection will be 99% dependent upon the billing software chosen. It would be more useful to cover the criteria for selection of components rather than the choices available.

With regards to the other software selection, the criteria seems to be whether or not the component ships with Solaris 8. If part of the rationale in using software is that it is open source, then use the open source version! This would provide ready availability of security updates, software upgrades, and so on rather than having to wait for Sun to release patches. As with any vendor-related book, the coverage is focused on Solaris 8. Many service providers use one of the many BSD and/or GNU/Linux variants for some or all of their server operations. It would have been nice for an OS besides Solaris 8 to be covered. Of course, this is wishful thinking given that the book belongs to the "Sun Blueprints" series. Finally, the appendices vary in their usefulness.

In the final analysis, the book is worthwhile simply for the capacity-planning equations it contains. But as a "How to Set Up Your Own ISP" guide, it didn't meet my expectations.

As always, I look forward to your questions and comments!

symlinks and hard links don't belong in /etc

by Steve Simmons

Steve Simmons has been an active member of the UNIX system admin community for two decades, including stints as president of SAGE and chairman of LISA.

scs@di.org

They don't belong in several other places as well, but let's for the moment just talk about /etc and look at some problems they introduce.

There are an awful lot of OSes out there that do their startups in System V style. To be more specific, they have a lot of /etc/rcX.d directories, where X is a run level like 0–6 or S. These directories have files like K20lpd, which starts up lpd.

So to modify run level 2, you cd to /etc/rc2.d and edit the files, right?

Wrong, as we all (should) know. In Solaris, all the various rc*.d/K20lpd files are hard links. Change one, you change them all. Well, depending on what editor or change technique you use. Recently in sage-members someone posted a problem with a Perl script that changed files in place and would end up replacing symlinks with new files while leaving the file the symlink referred to untouched. But is that bad? The answer is, it depends on what he wanted.

Take System V style rc files as an example. In Solaris, there are various /etc/rcX.d directories, where X is the various run levels available. Inside both /etc/rc0.d and /etc/rc2.d are files like K20lpd, which starts the line printer daemon.

In both run-level 0 and run-level 2, the line printer daemon does exactly the same thing. Sure enough, the two K20lpd files are one file with multiple hard links. So when you change it in place, you change the line printer characteristics in both run levels.

And sometimes you're changing unexpected things as well. For example, in Solaris 2.6, K20lpd has four hard links. Only two are in /etc/rc*.d/K20lpd. The other two are elsewhere. We'll come back to that in a minute.

Things can start biting you quite hard at that point. Most of us are smart enough to do backups of a system configuration file before changing it. The method you use to do that backup will affect the result of your changes. I like to do this:

```
# cd /etc/rc2.d
# mv file file-orig
# cp -p file-orig file
# vi file
```

This preserves everything about the original file, including inode modification dates. If you decide to undo it later by doing

```
# mv file-orig file
```

you've restored the system as nearly as possible to its original state.

But when symlinks and hard links are involved, things get dicey. If file is a symlink to something else, the file you're editing is the local copy only, not the other copy in rc0.d. Is that what you intended? Well, did you mean to change all lpd performance or just the rc2.d performance? Did you? Odds are good you didn't think about it. Until you've been bitten a few times, anyway.

An alternative (and potentially simpler) method of doing the same preservation is

```
# cd /etc/rc2.d
# cp -p file file-orig
```

```
# vi file
```

But when you restore it, you'd better remember to do

```
# cp -p file-orig file
```

rather than

```
# mv file-orig file
```

because if hard links are involved, you're going to get different results. So be sure and do it the same way consistently. And you should make sure all the other admins you work with do it the same way.

Different editors can introduce new issues. Emacs often makes automatic backups. Does Emacs preserve the inode numbers, mod dates, etc., when you revert the file? Do you *know*? What about vim? Pico? All versions of those editors? And what about your local Emacs customizations?

And what if you keep your rc files under RCS or CVS? Do they do in-place restores for hard links? Symlinks? And what about rdisting?

After a while, the experienced sysadmin knows to be careful about mucking about with the rcX.d contents. Unfortunately, that's not the only place we have to deal with this sort of thing. /etc/termcap is a symlink on many systems. So is /usr/man. You'd better just get in the habit of watching for symlinks and hard links any time you edit a configuration file, right? Unfortunately, yes. This little issue – the presence of symlinks and multiple hard links in configuration directories – leads to a morass of potential problems. Simply put:

Symlinks and multiple hard links don't belong in configuration files.

Period. They should be avoided whenever possible. And they can be, usually via a mechanism that makes the system *more* maintainable, not less.

Solaris 2.6, bless its pointy little head, almost gets it right. In 2.6, K20lpd has four hard links. They are:

```
/etc/init.d/lpd
/etc/rc0.d/K20lpd
/etc/rc2.d/K20lpd
/etc/rc2.d/S80lpd
```

What's needed is to get rid of the last three copies and instead replace them with something obvious, such as this shell script:

```
#!/bin/sh
#
# Strip the directory from the path and leading XNN from the name
# and run the core script in the base directory.
#
SCRIPT=`basename $0`
STRIP=`echo "$SCRIPT" | sed 's/^[A-Z][0-9][0-9]//'`
BASE=/etc/init.d
if [ "$STRIP" = "$SCRIPT" ] ; then
      echo "'$SCRIPT' is not a properly formed RC script name. Skipping." 1>&2
else if [ ! -x "$BASE/$STRIP" ] ; then
      echo "'$SCRIPT' base executable ('$BASE/$STRIP') not found. Skipping." 1>&2
else
      "$BASE/$STRIP" $@
fi ; fi
```

Symlinks and multiple hard links don't belong in configuration files.

symlinks and hard links in system areas are a recipe for problems with system administration.

When you go to edit /etc/rc0.d/S80lpd and find that script, you've just been alerted that you're looking at a shared configuration. You still face the decision of whether to modify the master ($BASE/$STRIP) or copy it to the local dir and modify just that one, but now you can't avoid making a conscious decision rather than letting the editor determine where the chips will fall. And no matter how you backed it up, cp or mv or CVS or RCS, the right thing will happen when you restore.

The current implementation of Solaris rc*.d files was a conscious decision on someone's part. The author was seduced by the attractive solution of hard links, and we sysadmins get to live with it. The proposed solution above gives almost all the benefits of the current system, but without the potential for easy error.

Unfortunately, others aren't as easy to fix. A quick glance at /etc in Solaris 2.6 shows almost 70 symbolic links. Almost all have the same bad reason for existence: backwards compatibility to old OS versions. Some are sheer laziness on the part of the vendor. For example, any link from /etc to ../sbin is present only so that old shell scripts didn't have to be updated. That's 53 links in /etc/ that can simply go away. Similarly, another batch are files that moved to /etc/inetd or /var/adm more years ago than I care to remember. On my Solaris 2.6 box, cleaning up just the obvious ones would get rid of all but three of the symlinks at the top of /etc.

Some of the ones left are things the vendor might have less control over. /etc/termcap is a good example. It is a symlink to /usr/share/lib/termcap. Its original move was done for space-saving reasons in /. By replacing it with a symbolic link, neither the OS vendor nor the third-party vendors had to make a change – even binaries continued working properly.

But how many vendors are still shipping those binaries to run on current systems? Zero, I'd wager. That's not to say the code has been fixed. With the backwards-compatible symlink in place, no one has any inducement to track down those legacy references to /etc/termcap and fix them. This would mean work even for system admins, whose users might have TERMCAP=/etc/termcap in various shell initialization files. But once those are fixed, those same admins will never have to go back and repair the broken links or mangled backup copies.

Frankly, symlinks and hard links in system areas are a recipe for problems with system administration. IMHO they are present only because somebody didn't bother to fix the *whole* system when making an improvement. We have been living with the unfinished 1% (and the problems they cause) ever since. It's time to fix it. We admins should fix our legacy user definitions (e.g., TERMCAP); that's easy enough to do.

The problem with symlinks and hard links is easy to state. We need to state it to the vendors and back it up with sample fixes such as provided above. Individual admins can move this problem along by complaining to the OS vendors. SAGE can help by stating the problem and carrying proposed solutions as the collective voice of the admins.

Fixing the third-party vendor issues are harder. But most OS vendors have existing processes by which they sunset certain OS features. We need to identify things such as /etc/termcap that need to be placed into that sunset process and push to get them there. Assuming it's not required by POSIX, there's no reason that /etc/termcap should have persisted this long. Simple recompiles should fix that particular problem; if we can get the link removals into the sunset processes, we'll eventually seen cleaner systems out there.

It can be done better, it should be done better. But it won't unless we complain about it.

remote monitoring with SNMP

A Practical Example

by Jozef Skvarcek

Jozef Skvarcek is currently working as a system administrator. He holds a PhD in Physics. Computer technology and science are his long-time hobbies.

jozef@photonfield.net

Introduction

My previous article (*;login:* Vol. 27, No.5, October 2002) introduced the Simple Network Management Protocol (SNMP) as a valuable tool in remote network monitoring. The configuration of the agent was shown using the NET-SNMP open source SNMP software as an example. The functionality of the protocol was demonstrated by retrieving the value of the system.sysUpTime OID and the values of the "host" branch, which could be exploited for measuring the disk space utilization. In this article we shall provide a detailed procedure for the monitoring of the operating system parameters.

Our objective is to monitor the system availability, disk and swap space utilizations, running processes, and system load. The system availability check will be performed by ping. A system will be considered available if ping succeeds, but if ping fails no further SNMP polls will be done. The monitoring of the parameters will be done by the SNMP polls. It will be assumed that the management station and the monitored systems have the SNMP agents properly installed and configured. The following examples will use the commands that are part of the NET-SNMP agent software. The management information contained in the Host MIB, defined by RFC 1514, will be assumed to be supported by the agent. In a real situation, we would probably have a number of identical machines that are to be monitored. Therefore, the script which will implement the monitoring procedure should provide means for specifying the common thresholds for the groups of identical systems. The script should issue a notification whenever the threshold value for any parameter is exceeded. The syslog will be employed for the notifications.

We would like the syslog messages generated by the script to go to their special log file, say, /var/log/snmp-monitor.log. In order to achieve that, we will use the local syslog facility local1, which should not be used for any other purpose. Therefore, we put into the syslog's configuration file /etc/syslog.conf the following line:

```
local1.*  /var/log/snmp-monitor.log
```

Of course, this is just an example. We could also direct the notifications to a remote syslog server, emit email or page messages, or send SNMP traps to a full-featured network management station (NMS) such as IBM Tivoli Netview.

Monitoring Disk Space Utilization

We would like to be able to measure the disk space usage on a remote system. How do we start? First, we need to determine the OIDs which hold the suitable management information. The Host MIB can be used for that purpose – it specifies the OIDs under the iso.org.dod.internet.mgmt.mib-2.host branch. By reading RFC 1514 we find hrStorageTable table which includes the entries relevant for achieving our goal. Among the others it contains the names of the file systems and their total and used sizes. Next, we want to see the exact result from the SNMP query of the hrStorageTable on the agent. In the following examples, we will work on the management station called "Jupiter" and will poll the agent named "Europa". On Jupiter we run

```
$ snmpwalk europa .iso.org.dod.internet.mgmt.mib-2.host.hrStorage.hrStorageTable
```

which produces many lines of output. Here is a truncated example useful for further discussion:

```
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageIndex.3 = 3
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageType.3 = OID: host.hrStorage.hrStorageTypes.hrStorageFixedDisk
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageDescr.3 = /var
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageAllocationUnits.3 = 4096 Bytes
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageSize.3 = 1741346
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageUsed.3 = 22976
```

It describes the /var file system as given by the hrStorageDescr OID. The OID instance corresponding to /var is the number 3 which identifies the other OIDs that belong to the same file system. The information is sufficient for calculating the percentage of the free space in /var using the formula

```
Free space = ( 1 - hrStorageUsed / hrStorageSize ) * 100%.
```

The result can be compared to a threshold value; if it is lower, a notification should be generated.

Monitoring Swap Space Utilization

The approach is very similar to the previous paragraph. Again, we can take advantage of the information from the hrStorageTable from the Host MIB. However, now we are interested in the different lines from the output from the snmpwalk command:

```
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageType.102 = OID: host.hrStorage.hrStorageTypes.hrStorageVirtualMemory
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageDescr.102 = Swap Space
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageAllocationUnits.102 =1024 Bytes
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageSize.102 = 264952
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageUsed.102 = 0
```

These lines report the values of the OIDs with the instance number 102, which belongs to Swap Space. Similarly as in the previous case we can calculate the free space and compare it to the threshold.

One of the arguments for implementing SNMP for the remote monitoring is the fact that it is an open standard which allows for monitoring different operating systems or hardware platforms. The procedure described here is exactly like that. Well, almost exactly . . . What is our point? There are small nuances in the management information provided, even by the same agent software, depending on the particular platform. For example, the NET-SNMP agent on Solaris 8 does not show Swap Space in the hrStorageTable. On the other hand, it shows the /tmp file system with the storage type of a fixed disk. We know that the size of /tmp on Solaris is correlated with the amount of the available swap; therefore, as a workaround we could monitor the free space in /tmp. The conclusion is that one "has to see" the content of the management information before writing a monitoring application. In most cases, there are multiple ways to meet the objectives.

Monitoring CPU or System Load

The CPU load can be observed by checking the values of the OIDs in the hrProcessorTable table defined in the Host MIB. The relevant OID is named hrProcessorLoad. However, according to our experience this information is not provided by the NET-SNMP agent (v4.2.3) on RedHat Linux 7.2 or Solaris 8, an example of the kind of peculiarity mentioned in the previous paragraph. What can be done if we have GNU/Linux or Solaris agents? We can monitor the system load instead. The CPU and the system loads are not the same thing but they are normally correlated. That fact makes them equivalent for our purpose. The system load can be measured by polling the OIDs in the NET-SNMP MIB. The MIB file is called UCD-SNMP-MIB.txt and it is included with the NET-SNMP sources. The MIB defines the objects under the .iso.org.dod.internet.private.enterprises.ucdavis branch. By reading the MIB file, we locate the relevant information in the laTable table so that we can run, on Jupiter,

```
$ snmpwalk europa .iso.org.dod.internet.private.enterprises.ucdavis.laTable
```

which returns

```
enterprises.ucdavis.laTable.laEntry.laLoad.1 = 0.06
enterprises.ucdavis.laTable.laEntry.laLoad.2 = 0.03
enterprises.ucdavis.laTable.laEntry.laLoad.3 = 0.01
```

From the MIB, we know that the three instances hold load averages over one, five, and ten minutes. The five-minute average (instance 2) might be a good candidate for our purpose since it smoothes out occasional spikes. The value can simply be compared to the threshold, and if it is higher, then a notification should be created.

Monitoring Running Processes

Each production system runs specific processes that are important for the functionality of the environment it is a part of. For example, a Web server may run Apache. If Apache is not running, then we would like to be notified. Once again, the Host MIB comes in handy since it includes the hrSWRunTable, which contains useful objects that will allow us to meet the objective. Let's see exactly what information we can get by running the following on Jupiter:

```
$ snmpwalk europa .iso.org.dod.internet.mgmt.mib-2.host.hrSWRun.hrSWRunTable
```

Since the output is long, here is a truncated version for the purpose of illustration:

```
host.hrSWRun.hrSWRunTable.hrSWRunEntry.hrSWRunIndex.535 = 535
host.hrSWRun.hrSWRunTable.hrSWRunEntry.hrSWRunName.535 = "syslogd"
host.hrSWRun.hrSWRunTable.hrSWRunEntry.hrSWRunPath.535 = "syslogd"
host.hrSWRun.hrSWRunTable.hrSWRunEntry.hrSWRunParameters.535 = "-m 20 -r"
host.hrSWRun.hrSWRunTable.hrSWRunEntry.hrSWRunType.535 = application(4)
host.hrSWRun.hrSWRunTable.hrSWRunEntry.hrSWRunStatus.535 =  runnable(2)
```

This tells us that the syslogd daemon is running with the given parameters. If we specify the name of the process that should be present on the system, the script will try to find a match among the running processes. If the match is not found, the script will create the notification.

The Perl script snmp-monitor.pl which is available at *http://www.photonfield.net/snmp-monitor.html,* implements the algorithm for measuring the disk space discussed above. The script is executed on the management station. Our management station runs RedHat Linux 7.2 and our agents are a mix of Solaris, GNU/Linux. Very few modifications would be necessary for porting the script to another platform.

The script works with SNMPv3 using the set up described in my October *;login:* article. If SNMPv3 is not supported by the agents — for example, if there are Windows 2000 clients — then the SNMPv2 can be utilized instead. In that case we should modify the $SNMPWALK and $SNMPGET variables. The script parses the configuration file specified by $CONF_FILE. The configuration file has its own specific format:

```
disk        <disk_fs>     <disk_free>
host        <agents>
endprofile
```

The disk line is for specifying disk usage thresholds, <disk_fs> (string) is the name of the file system, and <disk_free> (integer) is the minimum amount of the free space expressed in percentages. There can be multiple disk lines. The purpose of the host line is to specify the hostname of the agent we want to monitor. There can be multiple host lines so that the identical parameter's thresholds can be used for monitoring a number of agents. The endprofile line specifies the end of a "profile," which is a set of related disk and host entries. We can have many profiles in the configuration file. The comments are allowed and should be on the lines that begin with #. For example, the following configuration file will monitor the file systems / and /var on agent Europa. The minimum amount of free disk space in both file systems is 20 percent.

```
disk    /       20
disk    /var    20
host    europa
endprofile
```

The script populates the arrays @disk_fs, @disk_free, and @agents according to the values found in its configuration file. The elements of the arrays are passed as the arguments when calling the subroutine &check_disk. The SNMP polls are performed by calling the binaries that are part of the NET-SNMP agent. The logger utility is employed for logging the notifications.

The management information available through SNMP is wide and is defined by a large number of standard or vendor-specific MIBs.

The algorithms for measuring the swap space and system load and for monitoring the running processes are very similar to the disk space monitoring. Their practical implementation is left to the reader.

Conclusions

The approach to utilization of SNMP for remote monitoring has been discussed and an illustrative Perl script has been shown. The development requires a certain degree of flexibility because the amount of management information can vary slightly on different platforms or for different SNMP software. The presented script has simple functionality. Nonetheless, it provides us with the single point of administration of the monitoring operations. All the monitored parameters are defined in the single configuration file.

The script could be enhanced in many ways. For example, it has no "memory," which means that it issues a notification every time it finds an exception. We may want to receive only one notification when the exception is detected for the first time and then another one when the things get back to normal. In addition, the script is serial, executing the pings and the SNMP polls one after another. If the number of agents is large it may take many minutes to poll all of them, especially when there are some timeouts. In order to make the execution time shorter we could make the polls run in parallel for multiple hosts, or perhaps we could develop a multi-threaded version.

The scope of the possible monitored targets is not limited to the presented parameters in any way. The management information available through SNMP is wide and is defined by a large number of standard or vendor-specific MIBs. Our ambition was to give readers a helpful practical example which would provide them with a head start for solving their particular problems.

References

Script

http://www.photonfield.net/snmp-monitor.html

Article

J. Skvarcek, "Remote Monitoring with SNMP," *;login:*, October 2002.

Books

D.R. Mauro and K.J. Schmidt, *Essential SNMP* (Sebastopol, CA: O'Reilly & Associates, 2001).

D. Perkins and E. McGinnis, *Understanding SNMP MIBs* (Upper Saddle River, NJ: Prentice Hall, 1997).

P. Simoneau, *SNMP Network Management* (New York: McGraw-Hill, 1999).

D. Zeltserman, *Practical Guide to SNMP v3 and Network Management* (Upper Saddle River, NJ: Prentice Hall, 1999).

Sites

NET-SNMP (UCD-SNMP): *http://net-snmp.sourceforge.net/*

RFC: *http://www.ietf.org/rfc.html*.

introduction to the border gateway protocol (BGP)

by Greg Hankins

Greg Hankins is a senior systems engineer with Riverstone Networks and has been designing and deploying service provider and enterprise networks for the past 10 years. An avid Linux user since early 1994, he is best known for his contributions to the Linux Documentation Project in many different roles.

gregh@twoguys.org

This article presents an introduction and overview of BGP, the routing protocol of choice for large-scale IP routing. BGP has gained a reputation of being somewhat of a black art (even blacker than the art of SCSI), and experienced BGP-savvy network engineers are still at a premium. It certainly is a complex protocol that cannot be thoroughly explained in any one article or even a small book, and takes time and experience to master. However, I hope that after reading this article you will at least have a fundamental understanding of the BGP protocol.

Background

BGP is an Exterior Gateway Protocol (EGP). EGPs were developed to provide inter-domain routing between networks called autonomous systems. An autonomous system (AS) is a set of networks and routers under common administration, which are assigned a globally unique number. EGPs have fundamentally different requirements from an Interior Gateway Protocol (IGP) such as OSPF, IS-IS or RIP. IGPs are run within your network to communicate reachability about networks under your control, while EGPs are run at your network border to provide reachability information about networks outside of your control.

Whereas IGPs were designed to scale to a few thousand routes, EGPs were designed to scale to huge numbers of routes and to provide routing policy mechanisms. EGPs primarily make routing decisions based on the path of networks to a particular destination, not on the hops within each of the individual networks the path traverses. Think of it this way: If you are driving across country, you want a map that shows you the interstate highways, not detailed maps of all the cities along the way.

BGP first became an Internet standard in 1989 and was originally defined in RFC 1105. It was then adopted as the EGP of choice for inter-domain routing. The current version, BGP-4, was adopted in 1995 and is defined in RFC 1771. BGP-4 supports Classless Inter-Domain Routing (CIDR) and is the routing protocol that people use today to route between autonomous systems.

It has proven to be scalable, stable, and extensible, and it provides the mechanisms needed to support complex routing policies. When people talk about BGP today, they implicitly mean BGP-4. There is no need to specify the -4 version number because no one uses earlier versions, and very few vendors even still support them.

BGP continues to evolve through the Internet standards work in the IETF IDR working group; the latest draft version is at *http://www.ietf.org/internet-drafts/draft-ietf-idr-bgp4-18*. As the Internet routing requirements change, BGP is extended to continue to provide the knobs and dials needed to control routing information and to support new network requirements. The base RFC has been extended by several RFCs and I-Ds that define new features. Most recently, for example, BGP has been extended to provide support for building MPLS-based VPNs and a graceful recovery mechanism from router crashes.

BGP is the only protocol that is suitable for use between autonomous systems because of the inherent support for routing policies that the path attributes provide.

Protocol Details

We call BGP a path vector protocol because it stores routing information as a combination of a destination and attributes of the path to that destination. The protocol uses a deterministic route selection process to select the best route from multiple feasible routes using the path attributes as criteria. Characteristics such as delay, link utilization, or router hops are not considered in this process. We will see how BGP uses these path attributes later on.

Unlike most IGP protocols, BGP only sends a full routing update once when a BGP session is established and then only sends incremental changes. BGP only recalculates routing information relative to these updates; there is no regular process that must update all of its routing information like the SPF calculations in OSPF or IS-IS. Although IGP convergence may be faster, an IGP will simply not scale to support the number of routes needed for inter-domain routing. IGPs also lack the path attributes that BGP carries, which are essential for selecting the best route and building routing policies.

BGP is the only protocol that is suitable for use between autonomous systems because of the inherent support for routing policies that the path attributes provide. These policies allow you to accept, reject, or change routing information based on the path attributes before such information is used to make forwarding decisions. This gives network operators the ability to control routing information according to their particular needs, including rejecting routing information they might not want. Neither OSPF or IS-IS provide policies to reject or change routing information and thus should not be run between autonomous systems. RIP provides such policies, but suffers from even greater scalability issues.

BGP runs in two modes: EBGP and IBGP. EBGP (Exterior BGP) is run between routers in different autonomous systems, and IBGP (Interior BGP) is run between routers in the same autonomous system. It is necessary to run IBGP between backbone routers in order to provide each of them with a complete view of the routing table. This allows traffic to take the best exit point out of your network.

Protocol Mechanics

BGP uses TCP to establish a reliable connection between two BGP speakers, or peers, on port 179. Exactly one TCP session is established between each peer for each BGP session. No routing information can be exchanged until the TCP session has been established. This implies that each BGP speaker must have working IP connectivity between them first, which is usually provided by a directly connected interface or the IGP.

Since it uses TCP, BGP does not need to worry about transport issues such as data sequencing or fragmentation. TCP takes care of these problems and simply hands BGP a reliable pipe for transporting its messages. For added security, MD5 signatures can be used to authenticate each TCP segment.

One definition is needed before we look at the protocol in more detail. An *IP prefix* is simply an IP network with its mask: for example, 10.0.0.0/8. It is technically incorrect to call this an IP route as it pertains to BGP because the prefix only specifies the network and mask, not how to reach it.

MESSAGES

BGP uses five defined message types to communicate its routing information. You don't need to know all of the details about them, but it is helpful to at least know each one and how BGP uses it. Each message uses a fixed header, with a variable type-and-length field. This allows multiple BGP messages to be sent within one TCP segment.

OPEN

The OPEN message is the first message that is exchanged between BGP peers after the TCP session is established. It contains each peer's configuration information and handles any negotiations on exactly which BGP extensions each peer supports. Only one of these is sent at the beginning of the session.

UPDATE

These messages carry the actual routing information. UPDATE messages are used to signal new routing updates and to withdraw old routing information. The IP prefix, along with the path attributes, is sent in these messages. BGP is very efficient about how it transmits the routing information. If multiple prefixes share the exact same path attributes, BGP will send multiple prefixes in an UPDATE message with one copy of the associated path attributes. UPDATE messages are sent as often as they need to be, but remember that BGP only sends a complete routing update at the beginning of the session. Then it only sends incremental changes.

KEEPALIVE

The KEEPALIVE message is simply a message that keeps the BGP session up, indicating that the router is still operating normally. A timer is reset each time a KEEPALIVE is received. If none are received within a predefined time period, the timer expires. At this point, the other router is presumed to be unreachable and the peering session is torn down.

NOTIFICATION

A NOTIFICATION message is used to communicate errors. All error types are predefined, making it very easy to troubleshoot a BGP peering problem. The NOTIFICATION message simply contains exactly what was wrong in form of an error code and an error subcode. After it is sent, the BGP session is closed.

ROUTE-REFRESH

The ROUTE-REFRESH message is not defined in the base BGP specification but as an extension to BGP. However, it has been so widely implemented that it only makes sense to mention this message here. This message is used to request a complete retransmission of a peer's routing information without tearing down and reestablishing the BGP session (remember, BGP only sends a complete routing update once).

Using this extension, routing policy changes can be made without storing an unmodified copy of the peer's routes on the local router, which in turn saves RAM and resources. If a change is made to the routing policies, then a route refresh is requested from the peer, causing the new policy to take effect.

The ROUTE-REFRESH message was designed to be protocol independent. Thus, for example, you can request a retransmission of a peer's IPv4 unicast routes but none of its IPv6 routes.

BGP is very efficient about how it transmits the routing information.

State Machine

BGP uses a Finite-State Machine with carefully defined events and state transitions. This allows BGP to know exactly what to do next, whenever anything happens. Briefly, the FSM starts out in the Idle state, then transitions through several states as a TCP connection is established, and options are negotiated. Finally, BGP reaches the Established state, and starts exchanging routing information in form of UPDATE messages. If any errors occur along the way, or malformed or invalid routing information is received, BGP shuts down the session and goes back to the Idle state. One FSM is maintained for each BGP session, allowing many peers to exist in different states.

Path Attributes and Routing Policies

No article on BGP would be complete without mentioning some of the path attributes that BGP uses to communicate details about each path to a destination. Though the details can be, well, detailed, I will provide enough to give you the general idea of each. Routing policies can be used to accept, reject, or even change path attributes on routing information that is sent or received between BGP peers. Routing polices are the keys that unlock extremely powerful control over routing information, which can be as granular or as coarse as you need it. For example, you can apply a policy to a single IP prefix (say, 192.168.0.0/24) or all 110,000 routes received from a particular peer.

Let's discuss the most common path attributes briefly, and how each can be used to control routing information.

AS Path

The AS Path is an ordered list of all autonomous systems that an IP prefix has traversed, from right to left. Each autonomous system is represented by an integer from 1 to 65535, and is assigned by the regional registry (ARIN, RIPE, APNIC, etc.). The shorter the AS Path, the more desirable it is. For example, if a prefix has the path "7018 3356 4355" we would know the following: it was originated by AS 4355, it traversed AS 3356, then AS 7018. If your router had two paths to the same destination that were "7018 3356 4355" and "1 4355", it would choose the second because it is a shorter path. You can influence how other networks reach an IP network by making the AS Path shorter or longer, which then makes it more or less desirable.

AS Path regular expressions can be used for matching in routing policies. This gives you a very powerful classification mechanism to make routing decisions. For example, say you wanted to black hole any IP networks owned by Microsoft. A simple regular expression that denies any prefix whose AS Path ends in 8070 (Microsoft's AS number) from entering the routing table can easily be applied.

Local Preference

BGP provides bi-directional metrics for selecting the best route. The Local Preference attribute is used to control how traffic leaves your network, and it is represented as a 4-byte integer. A higher Local Preference means a higher degree of preference. Using a clever combination of this metric, network operators commonly set up primary and backup egress paths. Local Preference tuning is also a popular way of load sharing transit connections.

Multi-Exit Discriminator (MED)

This path attribute is used to control how traffic enters your network. Though the AS Path can be used at a coarse level, MEDs provide finer control. A MED is represented

as a 4-byte integer. A lower number means a higher degree of preference, opposite of the Local Preference. MEDs can also be used to provide redundancy and load sharing, with one caveat: They are only compared between the same autonomous system. Because each network's policy is different, comparing MEDs among different autonomous systems would be like comparing apples and oranges, resulting in some very strange routing. In some cases, the IGP metric can be used as the MED, optimizing the ingress traffic flow even further.

COMMUNITIES

One of the hardest problems in BGP is selecting a few prefixes out of many. Selecting one or all is easy, but how do you choose 500 particular prefixes from 110,000? Communities are simply arbitrary tags that are associated with a prefix. Using communities is a popular way to tag certain prefixes for later matching in a policy. This type of tagging is extremely flexible and, most importantly, dynamic. By using communities, you don't need to rely on lists of IP prefixes that must be updated by hand every time a change in the network is made.

The classification possibilities are endless. For example, some operators assign all prefixes from the same geographic location to the same community. This allows them to make routing policy decision on, say, all networks that are in a particular city, or even continent, without knowing or caring exactly which IP addresses they might be.

When to Use BGP

One of the most important decisions to be made is whether to even run BGP. A lot of thought must be put into this decision, and you should weigh the benefits and drawbacks very carefully. Simply using static routes can save time and a lot of complexity.

As you can see, BGP is a complex protocol, and configuring the routers to run BGP is only a tiny step in implementing BGP in your network. Your network engineers and operators must understand the protocol in great detail in order to make correct design and implementation decisions and to maintain and troubleshoot the network. Additionally, you must understand how to build routing policies, as these are essential in making BGP do what you want. After all, BGP does exactly what you ask it to do, not what you mean it to do.

Here are some very simplified guidelines to help you determine if BGP is right for your network.

- If your network is single-homed to an ISP, you don't need to run BGP. Just use static routing between your network and the ISP for simplicity.
- If your network is multi-homed to one or more ISPs, you might need to run BGP. Again, if one or more static routes will work, each service provider can configure their routers so that traffic is shared between your transit links.
- If your network is multi-homed and you are designing your network for redundancy, load sharing, or want to optimize routing between your Internet transit links, you will need to build BGP routing policies to do this. In this case you need to run BGP with each ISP.

Further Reading and Resources

Unfortunately, many BGP topics were not covered in this article. BGP offers much more than what was discussed here, and we have barely scratched the surface of its capabilities. Hierarchy and scaling, capabilities such as authentication and graceful restart, and many other necessary details are all fascinating topics for further study.

Fully understanding all aspects of the protocol will give you the ability to design, deploy, and scale complex and resilient networks.

All related RFCs and I-Ds can be found on the IDR Working Group Web page (*http://www.ietf.org/html.charters/idr-charter.html*), and this is a good place to start reading if you are interested in the gory details. For added fun you can join the mailing list, where editorial changes and technical issues about the protocol are discussed.

Don't send operational questions to the IDR list though; it is strictly used for work related to developing the protocol itself. For technical questions, you are better off joining one of the many lists that are run by network operators for the purpose of discussing operational issues. A great list of mailing lists can be found on NANOG's ISP Resources page (*http://www.nanog.org/isp.html#lists*).

Additionally, two very good books have been written that cover BGP very nicely:

Bassam Halabi and Danny McPherson, *Internet Routing Architectures*, 2d ed. (Indianapolis: Cisco Press, 2000).

This book is still considered to be the BGP bible. It is an excellent and in-depth book, with many simple and complex practical examples. The configurations are Cisco specific, but the principles apply to any vendor.

John W. Stewart, *BGP4: Inter-Domain Routing in the Internet* (Reading, MA: Addison Wesley Longman, 1999).

A short and easy, vendor-neutral introduction and overview of BGP, Stewart's book does not have many practical examples, but it sure is great to keep handy as a BGP reference.

Finally, if you are interested in learning more about BGP there are many software implementations available for you to use. All you need is a PC to get started:

GNU Zebra (*http://www.zebra.org/*)

Zebra is a fully functional routing engine that runs on most UNIX systems. It supports BGP, OSPF, and RIP for IPv4 and IPv6. This one also features a Cisco-like CLI and is probably the best one to use for learning about routing protocols.

MRT - Multi-Threaded Routing Toolkit (*http://www.mrtd.net/*)

MRT is a freely available implementation that supports IPv4 and IPv6 routing protocols. BGP and RIP are supported. It runs on most UNIXes and MS Windows, too.

GateD (*http://www.gated.org/*)

GateD provides a full implementation of IPv4 and IPv6 routing protocols such as BGP, OSPF, IS-IS and RIP. GateD code is available at no cost to universities and research institutions. Commercial users must pay for a license.

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 Editorial Director at Matrix.net. He owns neither a dog nor a cat.

<peter@matrix.net>

While I was at LISA in Philadelphia, Rob Kolstad asked me what I might suggest as a good system administration textbook. I told him I had no idea, but that I'd think about it.

I then realized that I'd received a large number of sysadmin books over the past year, and that I might as well put much of the material together here.

Oldies

Over the years, I've had two standbys: Frisch and Nemeth et al. Now there are two Nemeths: UNIX and Linux. At the same time, Æleen Frisch has come out with a new edition.

Nemeth et al., 3d edition (in royal purple) may be my favorite sysadmin book. Period. As Eric Allman has said, "This is not a nice, neat book." We don't live in a nice neat world, after all. Frisch's 3d edition comes in a close second. It now weighs in at over 1100 pages. I keep both of these on hand: I look things up in them and consider them irreplaceable.

But this doesn't answer Rob's query: I certainly wouldn't use either of these as a textbook for a course.

New Creations

There are three new books to consider, each of them looking like a textbook, with exercises after each chapter. At the most elementary end of the scale is Dave Taylor's *Teach Yourself . . .* volume. I don't think you can learn to be a sysadmin in 24 hours, but you can learn a great deal. And this book is really right for someone who's got no previous experience as an administrator. If you're running Linux, Solaris, MacOS X on a desktop or a laptop, this book will suit you. If you're in a commercial environment, it's not enough. Limited, but quite good.

Limoncelli and Hogan is a truly outstanding book. I wrote about it last year, and this issue of *;login:* carries a full review by Robert D. Haskins.

Freeland and McKay is the "highest" level among these books. I see it as focused on the needs of the in-service administrator. They offer all sorts of advice and don't worry too much about any individual system. I liked the detailed explanations of many topics. And these are high-level elucidations, not the simpler ones you'll find in Taylor.

Rob, here's your answer: If you need a textbook, I'd go with Limoncelli and Hogan for the basic course, Freeland and McKay for the advanced course.

Frisch and Nemeth are the standby references.

Linux

Nemeth et al. is the overall winner for Linux administration: It's intelligent, full of insights, and looks at the implementation of concepts. My copy lives at home, where I run Linux on my desktop and on my laptop.

Linux Security

I was critical of parts of Bob Toxen's book two years ago (Feb. 2001). The new 2d edition is an improvement on what was a good book. I've gradually become more and more interested in security issues, and this is an excellent piece of work on hardening a Linux system. The CD-ROM contains both a bunch of open source tools as well as a number written by Toxen.

BOOKS REVIEWED IN THIS COLUMN

ESSENTIAL SYSTEM ADMINISTRATION, 3D ED.

ÆLEEN FRISCH

Sebastopol, CA: O'Reilly, 2002. Pp. 1149. ISBN 0596003439.

TEACH YOURSELF UNIX SYSTEM ADMINISTRATION IN 24 HOURS

DAVE TAYLOR

Indianapolis, IN: SAMS, 2002. Pp. 528. ISBN 0672323982.

THE PRACTICE OF SYSTEM AND NETWORK ADMINISTRATION

TOM LIMONCELLI & CHRISTINE HOGAN

Boston, MA: Addison-Wesley, 2001. Pp. 776. ISBN 0201702711.

THE COMPLETE SYSTEMS ADMINISTRATOR

CURT FREELAND & DWIGHT MCKAY

Lawrenceville, NJ: Delmar Learning, 2002. Pp. 880.ISBN 0766835197.

LINUX ADMINISTRATION HANDBOOK

EVI NEMETH ET AL.

Upper Saddle River, NJ: Prentice Hall, 2002. Pp. 928.ISBN 0130084662.

REAL WORLD LINUX SECURITY, 2D ED.

BOB TOXEN

Upper Saddle River, NJ: Prentice Hall, 2002. Pp. 848 + CD-ROM. ISBN 0130464562.

BACKUPS AND RECOVERY

W. CURTIS PRESTON AND HAL SKELLY

Berkeley, CA. SAGE – The System Administrators Guild, 2002. Pp 72. ISBN 1-931971-02-1

DESIGNING SYSTEMS FOR INTERNET COMMERCE, 2D ED.

G. WINFIELD TREESE & LAWRENCE C. STEWART

Boston, MA: Addison-Wesley, 2002. Pp. 496. ISBN 0201760355.

PROGRAMMING IN THE .NET ENVIRONMENT

DAMIAN WATKINS ET AL.

Boston, MA: Addison-Wesley, 2002. Pp. 560. ISBN 0201770180.

Backups

I have a love-hate relationship with backups. I hate doing them, I hate storing them, and I really rely on them on the (rare) occasion when a recovery is needed.

Preston and Skelly have written a neat 72 pages in the SAGE "Short Topics" series. If you're reading this, you deal with information. If you deal with information, you need to store it and be able to recover it. My only cavil is that all the references are to URLs. Surely, there are some printed works that might have been cited.

Commerce

I liked the first edition of Treese and Stewart five years ago. The new edition is nearly 100 pages thicker, but worthwhile. The (new) sections on XML, content provision, and Web services are very good, indeed.

Humor?

When I received Watkins et al., I thought it was a joke. After I began reading it, I realized it was a farce – perhaps a tragicomedy. If you think that Microsoft is poised to take over the universe, this may serve as a piece of the road map. Outside of an appendix on an aborted version of Perl, there's no language that I use listed. Those of you who know VB or C# or Oberon may find this book useful. As I find the notion of programming in the .NET environment ludicrous, I consider this book less than worthless. The perfect gift for the Microsoft sycophant.

Sorry.

[See also p. 41 for two additional reviews by Robert Haskins - ed.]

USENIX and SAGE Need You

People often ask how they can contribute to our organizations. Here is a list of tasks for which we hope to find volunteers (some contributions not only reap the rewards of fame and the good feeling of having helped the community, but authors also receive a small honorarium). Each issue we hope to have a list of openings and opportunities.

The SAGEwire and SAGEweb staff are seeking:

- Interview candidates
- Short article contributors (see *http://sagewire.sage.org*)
- White paper contributors for topics like these:

Back-ups	Emerging technology	Privacy
Career development	User education/training	Product round-ups
Certification	Ethics	SAGEwire
Consulting	Great new products	Scaling
Culture	Group tools	Scripting
Databases	Networking	Security implementation
Displays	New challenges	Standards
E-mail	Performance analysis	Storage
Education	Politics and the sysadim	Tools, system

- Local user groups: If you have a local user group affiliated with USENIX or SAGE, please mail the particulars to *kolstad@sage.org* so they can be posted on the Web site.

;login: always needs conference summarizers for USENIX conferences. Contact Alain Hénon, *ah@usenix.org*, if you'd like to help.

USENIX news

USENIX MEMBER BENEFITS

As a member of the USENIX Association, you receive the following benefits:

Free subscription to *;login:*, the Association's magazine, published six times a year, featuring technical articles, system administration articles, tips and techniques, practical columns on security, Tcl, Perl, Java, and operating systems, book and software reviews, summaries of sessions at USENIX conferences, and reports on various standards activities.

Access to *;login:* online from October 1997 to last month <*www.usenix.org/publications/login/login.html*>.

Access to papers from the USENIX Conferences online starting with 1993 <*www.usenix.org/publications/library/index.html*>.

The right to vote on matters affecting the Association, its bylaws, election of its directors and officers.

Optional membership in SAGE, the System Administrators Guild.

Discounts on registration fees for all USENIX conferences.

Discounts on the purchase of proceedings and CD-ROMS from USENIX conferences.

Special discounts on a variety of products, books, software, and periodicals. See <*http://www.usenix.org/membership/specialdisc.html*> for details.

FOR MORE INFORMATION REGARDING MEMBERSHIP OR BENEFITS, PLEASE SEE

<*http://www.usenix.org/membership/membership.html*>

OR CONTACT

<*office@usenix.org*>

Phone: 510 528 8649

30 Years Ago in UNIX

by Peter H. Salus

USENIX Historian

peter@usenix.org

1973! In February, Third Edition appeared. E.N. Pinson's name was added to the list of contributors. And Ken Thompson had added this to the front matter:

> "the number of Unix installations has grown to 16, with more expected."

That front matter also contained the (then) startling statement:

> "The three principal languages in Unix are assembly language (see as(1)), FORTRAN (see fc(1)), and C (see cc(1))."

C? C! A glance at cc(1) told you that it was a C compiler and referred you to the "C reference manual." It would be years until Prentice Hall published *The C Programming Language*.

At nearly four years of age, UNIX was getting ready to escape from New Jersey.

Actually, it had escaped the previous year – but it had not gotten far from home: only as far as AT&T in Manhattan, where Neil Groundwater was employing it to track wiring changes.

Explosive growth was yet to come. But it was lurching closer.

Summary of the USENIX Board of Directors Actions

by Tara Mulligan

Executive Assistant

tara@usenix.org

The following is a summary of the actions taken by the USENIX Board of Directors from June 13, 2002 to November 5, 2002.

Finances

The first draft budget for 2003 was discussed, with the following actions decided upon:

Membership: USENIX dues will be increased by $10 in each category.

USENIX will publish six issues of *;login:* in 2003, as in 2002 (vs. 7-8 in the previous 2 years).The USENIX staff was reduced by four full-time employees in 2002, and staff dedicated to SAGE was reduced by one.

USENIX will continue its membership in the Computing Research Association in 2003.

USENIX will support the NordU conference with a $5,000 guarantee, to be used toward the conference. If there is net revenue, the USENIX contribution will be repaid.

The Board approved a new structure for supporting membership categories, which will be $2500 for either a USENIX or a SAGE supporting membership; and $3500 for a Dual Supporting membership.

The $50 discount for registering for conferences via the web will continue in 2003.

Expenditures for Student Programs will change as follows:

Applications for the Student Research Grant and Scholars program will not be solicited in 2003.

The budget for Student Stipends to attend USENIX conferences was reduced for the final 5 conferences of 2002. This program will be funded in 2003 in the amount of $25,000, plus any outside donations.

Grants

USENIX will again be a sponsor of USA Computing Olympiad, in the amount of $15,000, which is a 50% reduction in funding from 2002.

Conferences

Middleware Conference. It was agreed to have USENIX co-sponsor the Middleware 2003 conference with ACM and IFIP, with no financial commitment.

HotOS Workshop. It was agreed that USENIX will take over organizing the 9th Workshop on Hot Topics in Operating Systems in cooperation with IEEE TCOS.

PKI Workshop. It was agreed that USENIX will co-sponsor the 2nd Annual Workshop, with no financial commitment.

The next Virtual Machine Research and Technology Symposium will be held in the Spring of 2004. USENIX will no longer use "Java" in the title of this event.

SAGE Certification

The SAGE Certification program's goals were not achieved. It was recommended that the testing be put on hold as the certification board seeks sponsorship to fund testing and the development of the m-SAGE test.

Thanks to Our Volunteers

by Ellie Young

Executive Director

ellie@usenix.org

USENIX's success would not be possible without the volunteers who lend their expertise and support for our conferences, publications, member services, SAGE. While there are many who serve on program committees, coordinate the various activities at the conferences, work on committees and contribute to this magazine, I would like to make special mention of the following individuals who made significant contributions in 2002:

The program chairs for our 2002 conferences:

- Darrell Long, First Conference on File and Storage Technologies
- Sam Leffler, BSDCon02
- Sam Midkiff, 2nd Java VM Research & Technology Symposium
- Carla Ellis, 2002 USENIX Annual Technical Conference
- Dan Boneh, 11th USENIX Security Symposium
- Ted Ts'o for organizing the 2002 Linux Kernel Developer's Summit

- Peter Honeyman, 5th Smart Card Research & Adv. Applications Conference (CARDIS '02)
- Alva Couch, 16th LISA Conference
- Jeff Mogul, 2nd Worshop on Industrial Experiences with Systems Software
- David Culler and Peter Druschel, 5th Symposium on Operating Systems Design & Implementation

The conferences' Invited Talk/Special Track Chairs:

- Chris Demetriou, 2002 Freenix Program Chair
- Dan Wallach for the invited talks at the 11th USENIX Security Symposium
- Matt Blaze and Ted Faber for the invited talks at the USENIX Annual Tech Conference
- Esther Filderman and Strata Rose Chalup for the invited talks at LISA
- David Williamson and Lynda True for coordinating the network and security tracks at LISA
- Lee Damon for organizing the "Guru is In" Sessions as USENIX Annual Tech and LISA

Esther Filderman for her efforts in organizing the AFS workshops at recent USENIX conferences.

Dan Geer and Andrew Hume for their 8 years of service on the USENIX Board of Directors (1994-2002)

Peter Honeyman, Mike Jones, John Gilmore, Jon "maddog" Hall, Kirk McKusick, Avi Rubin, Lois Bennett and Tina Darmohray for their service on the USENIX Board in 2002.

Peter Honeyman for his continued efforts in reaching out to international and other groups, e.g., OpenAFS community, SANE conference, EuroBSDCon, PKI workshop, Smartcards/CARDIS, and Middleware conference.

Rob Kolstad and Don Piele for their efforts as coach and director in making the USA Computing Olympiad which USENIX sponsors a success.

Andrew Hume for serving as liaison to the Computing Research Association.

The SAGE Executive Committee members for their service: Bryan Andregg, David Parter, Trey Harris, Gabriel Krabbe, Tim Gassaway, Geoff Halprin, Josh Simon

Andrew Hume, Jon "maddog" Hall, and David Parter for serving as liaisons for the USENIX Board and SAGE Exec committee.

The following people who served on the SAGE Certification Board:
Nancy Anheier, Lois Bennett, Jim Corder, Trey Harris, Andrew Hume, Mark Langston, John Stoffel, and J. D. Welch

SAGE Election Results

by Rob Kolstad

SAGE Executive Director

kolstad@sage.org

The results of the 2002 SAGE Executive Board election have been tabulated. Of 3,708 eligible voters, 1,055 cast ballots (a very high 28.45%). Voters cast an average of just under 5 votes each, suggesting that "power-voting" was a commonly used strategy.

Congratulations to these candidates who have won seats on the SAGE Executive Board:

Votes	Candidate
694	Trey Harris
668	Peg Schafer
517	Geoff Halprin
489	David Parter
450	Bryan Andregg
400	John Sellens
372	Gabe Krabbe

They took office at an in-person meeting held on February 1st and elected officers from among the group.

The following candidates did not win a seat on the Board:

Votes	Candidate
343	Jim Hickstein
321	Josh Simon
281	Matthew Barr
221	Curt Freeland
195	Luke Kanies
164	David Torrey
151	Gus Hartmann

It takes courage to stand for elections, especially one in which 50% of the candidates are guaranteed to lose. Please join me in congratulating all the candidates.

SAGE membership includes USENIX membership. SAGE members receive all USENIX member benefits plus others exclusive to SAGE.

SAGE members save when registering for USENIX conferences and conferences co-sponsored by SAGE.

SAGE publishes a series of practical booklets. SAGE members receive a free copy of the latest booklet when they join SAGE, and they receive a 33% member discount on all SAGE booklets. In addition SAGE members can freely access the full texts of the booklets on the Web.

SAGE sponsors an in-depth annual survey of sysadmin salaries collated with job responsibilities. Results are available to members online.

The SAGE Web site offers a members-only Jobs-Offered and Positions-Sought Job Center.

SAGE sponsors members-only mailing lists. The archive of the SAGE members list is available on the Web for members only.

SAGE EXECUTIVE DIRECTOR

Rob Kolstad: *kolstad@sage.org*

SAGE MEMBERSHIP

office@sage.org

SAGE ONLINE SERVICES

list server: *majordomo@sage.org*
Web: *http://www.sage.org/*
http://SAGEwire.sage.org
http://SAGEweb.sage.org
http://www.sagecert.org

conference reports

This issue's report focusses on LISA XVI

Our thanks to the Summarizers:

for LISA '02
Josh Simon, who organized the collecting of the summaries
Abiodun A. Alao
Paul Anderson
David Berg
Robert Beverly
Kuzman Ganchev
Jim Hickstein
Rob Kolstad
Martin Krafft
Renuka Nayak
James O'Kane
Will Partain
Peg Schafer
J. D. Welch
Steve Wormley
Garry Zacheiss

LISA XVI

Sixteenth Systems Administration Conference

Philadelphia, Pennsylvania, USA

November 3–8, 2002

KEYNOTE

Scaling the Web: An Overview of Google (a Linux Cluster for Fun and Profit)

Jim Reese, Chief Operations Engineer, Google

Summarized by J.D. Welch

We all know, use, and love Google, but how do they make it work? In this engaging talk, Jim Reese explained how custom software, massive replication and expendable, commodity hardware have allowed Google to answer 150 million Web search queries a day.

The core technology that separates Google from other search services is the PageRank system developed by founders Larry Page and Sergey Brin while graduate students at Stanford University. This system aims to objectively rank Web content by popularity; according to Reese, "a page's importance is the sum of the aggregate importance of the pages linking to it," so a page linked to from the *New York Times* is given more weight than one linked to by a high-school newspaper. In addition to assessing popularity, hypertext analysis is used to quantify the importance of elements on a page (e.g., larger text is probably more important).

To get a sense of scale of Google's challenge – there are 3.8 billion pages and 256 million Web users, and 85% of them use search services. Given this, any single machine will always be too small for the task, so index and page data is divided up into pieces, called "shards," which are distributed across many machines and multiple data centers. Thus, traffic is scalable by replication; the index is read-only, so single failures aren't fatal but only reduce capacity.

To answer a query, the Web server (a custom package called gws) queries index servers, document servers (cached pages), and ad servers, in parallel, and keeps trying until it gets a response. Each query may involve a dozen or more servers, using whichever reply comes in the fastest (the average query time is .23 seconds). Before the query reaches a Web server, however, it passes several load balancers, both global and local, which use various methods (including round-robin and least connections) to choose which servers to query.

"El Cheapo" PCs are used to maximize reliability through replication. Fault tolerance is kept very simple; timeouts are in the milliseconds, and machines are restarted automatically and regularly polled for their status. Racks of machines are very dense, with 80 half-depth 1U boxes in each, along with paired switches, load balancers, and Gigabit uplinks to the routers. All disks are local (100–120Gb/machine); large fans are mounted atop the rack and heat is drawn from the space between the machines in the center of the rack. All the machines run a "Googlized" distribution of RedHat Linux as well as proprietary tools for serving content and system monitoring.

For comparison with the new, very organized racks, Reese showed photos of historical configurations, including a custom-built 1U machine with four motherboards, eight disks, eight NICs, and one power supply, which was configured with the disks mounted over the processors separated by a sheet of Plexiglas (!).

INVITED TALKS

SECURITY ON MACOS X

John Hurley, Apple

Summarized by J.D. Welch

Hurley began by saying that Apple is in an interesting position to deal with security issues, as they manufacture the hardware, firmware, operating system, and often the end application, so a great degree of integration is possible in OS X security features.

Since OS X is based on BSD, many of the OS X security tools are ports of standard UNIX tools, oftentimes GUIfied with a Cocoa (native OS X Objective-C framework) front end. For example, the Sharing and Firewall control panels are a front end to ipfw. OS X also offers Kerberos, OpenSSH, OpenSSL, and other familiar UNIX tools in its default installation. Obviously, the use of familiar, often open source, packages is a departure from (and significant improvement over) OS 9.

A primary goal in designing the OS X security architecture was to make it easy to use these important features. Additionally, although many tools are presented plainly for users, they are configurable beyond what most users would bother with – good news for longtime UNIX users and security types. Also, Software Update encourages users to keep up-to-date with patches, as it automatically polls for and delivers updates directly to end users.

OS X implements a Common Data Security Architecture API, which provides an expandable set of crypto algorithms to various applications, including the Keychain (encrypted user information store) and Disk Copy (which can encrypt disk volumes). These "layered services" include file signing and certificate management as well as APIs for adding plug-in modules for additional services. With this modular architecture, developers can make use of security services without having to know a great deal about the specific component or service at work.

OS X makes a point of separating authorization from authentication, a move designed for next-generation applications, including smart card access, for which they are developing an SDK (called Smart Card Services) in collaboration with HP, Intel, and other vendors.

Out of the box, OS X is reasonably well locked down: Services like SSH, HTTP are off by default (but are easy to enable – from GUI or command line – if you know what you're doing), no ports are open, and the root account is disabled (sudo is used for administrative access). OS X honors UNIX user/group/file permissions and is designed to be a multiuser OS.

The Keychain is a cornerstone application, and was given much play in this talk. Accessible to all Cocoa, Carbon, and UNIX applications running under OS X, the Keychain provides an encrypted environment to store passwords for Web sites and file servers, encrypted disk volumes, and the like. Users "unlock" the Keychain with a master password, and applications can store and read data from the Keychain. Additionally, all Keychain items include an access control list for fine-grained control.

Another highlighted technology was the ability of Disk Copy, a utility available on all installations of OS X, to create encrypted disk images. Once the image is created, it can be mounted (with successful authentication) read/write, burned on a CD for transfer, etc.

The physical security of Apple hardware has also been considered. The XServe 1U rack mount server, for example, sends messages to the console when physical security of the rack is compromised; other Mac models can be "locked" with the Open Firmware Password, which prevents booting the machine without first authenticating, and prevents the use of startup commands (which can make the machine act like a FireWire disk or be booted into UNIX-permissions-free OS 9, for example).

This talk was a little marketing-heavy and didn't delve into technical details of the various systems implemented in OS X beyond their GUI expression, but it did provide a good introduction to the various services available to the OS X user or administrator.

ETHICS FOR SYSTEM ADMINISTRATORS: DILEMMAS FOR LISA 2002 ATTENDEES

Lee Damon and Rob Kolstad

Summarized by Steve Wormley

Unlike the medical profession, which has had thousands of years to develop ethical standards, system administration ethics are new. The mapping of conventional communications such as paper mail and the telephone do not work in the realm of email and Instant Messaging. The quantity of sensitive data online and issues such as identity theft contribute to awareness of the need for ethics and privacy guidelines in new technology.

Since computer ethics is a new area, novel situations and their attendant problems now happen at "Internet speed." We as system administrators need to have knowledge of ethics, privacy, and security so that we can protect rights and still get work done.

One definition of a professional is a person who conforms to the technical and ethical standards of a profession. For system administration to be regarded as a profession by the outside world, therefore, ethical standards need to be addressed.

A distinction should be made between ethics and policies. Since policies are well defined and generally not open to interpretation, establishing a site policy will often eliminate many ethical problems.

Ethics in the context of computer networks pertain to all privileged users, including anyone with access to others' information, even if that access is accidental; even help desk personnel, for example, need to be included.

Lee and Rob went on to present five scenarios involving ethical dilemmas for system administrators:

1. A project you worked on at a previous client had a flaw which could kill people if not corrected, but you only realized the flaw while at your current client, working on something similar, and could lose your job if you disclosed the flaw.
2. Your boss asked you to read the CTO's email to look for evidence of wrongdoing. You found a problem, reported it to your boss, and nothing was done. Now what do you do?
3. The third scenario was the often-repeated case where the boss wants the root password but is not competent on the system. How would you handle it?
4. In the course of routine administration you discover that your boss is discussing doing something evil, such as going to a competitor with customer lists. Now what?
5. You are providing network connectivity to a neighbor with children, and the children receive pornographic email. What do you do?

The Constitutional and Financial Arguments against Spam

Daniel V. Klein

Summarized by Martin Krafft

A trip to Dan Klein's home page (*http://www.klein.com*) reveals that he's a geek leaning toward the humorous. In his talk on the constitutional and financial argument against spam, he used exactly that tack. "Spam steals my time" could be seen as the motto as Dan proceeded to unroll his theories on preventing spam, keeping his audience focused while he delivered facts and ideas that, if nothing else, were entertaining.

He isn't a lawyer – he stressed this fact several times – so he approached his topic from a "common-sense" angle. Spam, or Monty Python's breakfast delicacy, is all those emails you never asked for – commercial mails advertising get-rich-quick schemes, mortgage loans, advertisements for penis enlargement devices, and other breathtaking new technology you wouldn't lack before or after the spam hit you. Dan started out by presenting a short history of his involvement with the Net and his exposure to spam, and then proceeded to lay out the numbers of an 80-day research period, in which he received one spam every 29 seconds. Even using a fairly restrictive set of anti-spam techniques, he claimed the ratio of ham to spam he receives is about the same as Earth's mass to Jupiter's. But to place his figures into relation to the real world, Dan quotes hotmail.com as being burdened by one billion spam messages per day.

He attacks the problem from two sides, starting with the constitutional. Freedom of speech seems to be commonly misunderstood and extended to argue for spam. Yet freedom of speech has exceptions (e.g., screaming "fire" in a public theater for no reason). You can say what you want, Dan pointed out, "but I don't have to listen to it, I can disagree, and [most importantly] you cannot make me pay for something I don't want to hear." Taken together with freedom of the press (I can print or refuse to print whatever I want, and so can you), and the constitutional argument against spam is right there: you are forcing your spam to be printed on my press, and I have no choice but to receive it. What Dan criticizes is that spammers seem to misinterpret freedom of speech as a guarantee of an audience, and freedom of press as a free method to print.

His financial argument against spam claims that spam costs the American people in the vicinity of 165 billion (!) dollars per year. In contrast to the 15 billion dollars made available to NASA every year and the 300 million the RIAA loses to "piracy" per year, this figure clearly indicates there's something severely wrong. Advertising isn't evil, it's necessary. Rather, the lack of regulation and control is what constitutes the problem. Spam is the cheapest method of advertising since it mostly raises costs for the recipients. It is marketing with a bullhorn, as Dan put it. He wants to "take back the Net."

Current anti-spam methods almost all come in the form of a filter-and-block setup on the recipient side. As effective spam filters are becoming more and more of a marketing technique of big ISPs like AOL and gmx.net, the voices around the "censorship" buzzword seem to be getting louder and louder. Censorship, in Dan's view, is ubiquitous rather than evil. "Abolish Censorship" may sound good but it reveals how little is known about the topic. People tolerate censorship more than they are willing to acknowledge, and yet scream at the idea of having someone filter their mail.

Dan sees little use in current methods, such as whitelists and confirmation systems. He wants a legal solution, and if not on a global level, then at least within the United States as a starter. However, he couldn't lay out a strategy for how such a law would be enacted and controlled, for which he isn't to blame — anti-spam is a challenge to the entire infrastructure and requires a lot of cooperation, from the MTA author to the ISP, from the government to the end user. He wants a global opt-out mechanism rather than one focused on individual advertisers.

Dan's talk, albeit very amusing, did not really offer anything new. Some audience members came to the talk to be comforted about their spam problems, others to get an idea of what spam is about. As such, Dan succeeded in reviving the subject and making it a prominent one, for a number of

anti-spam-related topics and discussions were evident throughout the remainder of the conference.

Risk-Taking vs. Management

Paul Evans

Summarized by Jim Hickstein

Paul gave a post-mortem of a dot-com company, Webvan, extrapolating from that experience to the broader view that social misperceptions of risk skewed business decisions and contributed to the dot-com bubble. He also looked at why our profession did not have enough credibility with management to influence those decisions.

The essence of capitalism is putting assets at risk in the service of profit. Professional financial managers get paid to balance the equation of assets, risk, and profit. But managers are people, and people tend to underestimate familiar risks and overestimate unfamiliar ones.

Unfamiliar risk abounded in the dot-com world, but it was asymmetrically distributed. Taking Webvan as an example, the grocery business is pretty well understood: Financial, operations, and even software development risks, were familiar. But risks in IT were unfamiliar to management. The "prevailing doctrine of risk" changed: In 1999, it was about not appearing above the fold of the *Wall Street Journal*; in 2000, the slogan became "five nines," whether merited or not.

The result was a business that overspent on redundancy (the larger perceived risk), while making fatal errors about the fundamental business model. Some people like to shop online, from a list and with a two-day lead time, but many others have a different, opportunistic style that only works in an actual store. Bad acquisitions, over-aggressive growth targets, and bad marketing decisions sank the company.

Paul gave several other examples of this misperception of risk, calling it peculiar to American society: the Challenger accident, the Persian Gulf war, the battle of Mogadishu. Yet several in the audience, from outside the US, thought it was not just an American trait.

Making Backups Easier with Disk

W. Curtis Preston, The Storage Group

Summarized by Renuka Nayak

The take-home message of W. Curtis Preston's talk was that system administrators should back up to inexpensive disks frequently while duplicating disk backups to tape. Doing anything else might lead to situations that SUCK (a mantra that was chanted throughout the interactive talk). The presentation was well-delivered and peppered with real-life examples that Preston had encountered throughout his career.

Preston first outlined some of the advantages, disadvantages, and challenges associated with tape drives. Tapes and tape drives are high speed and low cost, which makes them good archival solutions. But tape backups take a long time, and newer, higher-speed drives are becoming more expensive. Furthermore, it is difficult to make off-site tape copies with a stand-alone drive, which needs to swap tapes. When trying to access the tape in the drive, one might run into the problem of not having the desired tape in the drive. Challenges to using tape as the only backup medium include the time it takes to make tape-to-tape copies, the rigors of regularly perfuming full backups, the limitations on writing to a single tape drive from two shared servers, and the inability to know whether a tape is in good condition until you actually need to use it.

Using inexpensive disk arrays as a primary tool in backups in addition to using tape is an excellent way to address some of the challenges presented above, Preston suggests. There are IDE/ATA-based disk arrays that are addressable via Fibre Channel, SCSI, Firewire, NFS, and CIFS and which can use RAID configurations. These units are as low as $5,000 for off-shelf varieties, and it costs as low as $2,000 to build your own. Preston recommends buying enough disk for two full backups and many incremental backups. Then connect arrays to clients or backup servers and make backups. Finally, make duplicates (note that duplicating is different from backing up) of what is on your disk to tape. One might even want to place another disk unit off-site and replicate to it. Except in catastrophic disasters, one can easily restore from disk.

Preston then went on to say why using disk is better than using tape. Disk does not require a constant stream of data and neither is there the need to multiplex, as is the case for some tape drives. He claims that if disk backups are multiplexed, then the tape copies can be easily de-multiplexed without a performance penalty. Furthermore, since disk arrays can be protected via monitored RAID, the loss of a single disk would be monitored and repaired. Making disk-to-tape copies are easier than making tape-to-tape copies, and full backups can be performed less often, saving network and CPU utilization.

So, why should we even use tape at all? Preston argues that tapes are still good for archiving purposes so that older backups can be available. Tapes are also much cheaper than disk, allowing for multiple, stable copies to be stored "on the shelf" or off-site. Furthermore, tapes are not susceptible to file-system corruption, as disks may be.

To find out more information, email Curtis Preston at *curtis@thestoragemountain.com*.

"Who Are These People?" Internet Governance, Peering, and Legislation

Paul Vixie, Internet Software Consortium

Summarized by Robert Beverly

Mr. Vixie, a self-professed "graybeard" and "member of the loyal opposition," is a long-time programmer and maintainer of BIND (a software implementa-

tion of the Internet's domain name service). Mr. Vixie's talk explored some of the changing dynamics as the Internet metamorphoses from research network to commercial network to a component of national security. The talk was timely given the recent denial-of-service attacks on the root name servers.

Because of the academic nature of the early Internet, resources were given away freely, as needed, by a loose collection of individuals. Today many of these resources have become valuable commodities. Examples include IP address space, domain names, top-level DNS domains, autonomous system numbers, and protocol numbers. These shifts have produced a variety of stakeholders, all with different motives. The talk focused repeatedly on ICANN (Internet Corporation for Assigned Names and Numbers), a government-sponsored entity. An example of a current area of conflict is ICANN's control over the top-level DNS domains. Many Internet users feel that ICANN's policies toward new top-level domains is unjust. In fact, Mr. Vixie's contention was that because ICANN is a government-sponsored entity, it tries to be all things to all people and thus fails to serve anyone.

"The Government is coming and they want to take our toys." The operation of the root servers, so-called RSOs (Root Server Operators), is a clear example of a loosely organized resource that has become part of the critical infrastructure. How one becomes an RSO is a question with no answer today. Until Dr. Jon Postel's death, he alone made the determination. The original intent was to distribute the root name servers among commercial, research, and educational entities in different countries, such that there was enough natural distrust between operators to prevent a problem. No single entity should control, or be able to control, the entire system. Specifically, no single government should be able to take over the whole system. Today the root name servers are physically in the United States, England, Japan, and Sweden. As one of the long-term participants in the health of the global DNS system, Mr. Vixie is very concerned with current politics that may circumvent the original "graybeard" policies. Despite stating that everyone should be very concerned with recent policy directions, a general sense of pessimism emerged that those with "guns and money" would eventually prevail.

A second resource at stake is IP addresses. IP address space, once abundant, is now a valuable resource. An organization may have either provider-assigned space or provider-independent space. Smaller organizations requiring fewer addresses generally must obtain addresses from their providers. Provider-assigned space allows larger service providers to aggregate the routing announcements of their customers into a single aggregate. Limiting the total number of routes in the global Internet helps maintain its health and stability. The downside to provider-assigned space is that if an organization wishes to change providers, it must forfeit its current address space (which belongs to the provider) and obtain new space from its new provider. Obtaining new space requires renumbering the IP addresses for all of the machines the organization owns. Therefore, there is a large disincentive to switch providers, giving the existing large providers a distinct competitive advantage. Currently, obtaining provider-independent addresses requires providing justification to a regional registry for a minimum-sized block. The size of this minimum allocation is determined by the members of the registry who themselves are often network operators, creating an inherent conflict of interest.

Mr. Vixie noted that while current address-space policy prevents competition in many respects, the worst-case scenario is that government would take over the registries' duties. One would then have to go to the government to obtain IP address space, much as one would go to the government to get a business license. Mr. Vixie concluded the talk by warning of the dangers of people in decision-making positions who "don't understand the impact of those decisions." He urged attendees to get involved.

Nobody Notices Until It's Broken: Self-Marketing for Sysadmins

Moderator: Lee Damon, University of Washington

Panel: Karen Ken, Dan Klein, Strata Rose Chalup

Summarized by Abiodun A. Alao

The session was devoted to why system administrators are not very popular with other staff and why their point of view is difficult to convey. They are generally perceived as overpaid with unclear job schedules. It was noted that sysadmins make the first error in introducing their role. How do you explain what you do to someone who does not have any idea what the term "sysadmin" stands for? Here are some responses: "I work with computers"; "I make the Internet run"; "I manage computers."

It is essential that you are seen as a person, a member of the team and one whose contributions are valuable in the realization of the goals of the organization. We must also make people understand what we do in concise terms and in ways that they see how we contribute to their ability to meet their tasks. Avoiding all the techno jargons will go a long way toward making us understandable and more acceptable. In the alternative we may have to teach people to speak our language (even at the risk of training them to take over from us). Finally, we must also learn the language of business since we serve the business world.

Many of us complain we do not get any respect from coworkers. Can we do a better job at marketing ourselves? While it may not always be possible to make others see things from our point of view or even understand our role, a sysadmin

who takes time to explain what we do and why will be doing a good job and will help improve our image. If people are not savvy, make use of pictures. Often we are seen as capable of providing the silver bullet to all problems. When the system fails to meet the "expectations" of the customer, we are seen as incapable. When people do not understand or lack the capability to effectively use the IT solutions provided, their tendency is to blame the IT expert, especially the system administrator.

Granted, no matter how hard you try, you can't get everyone to cooperate or appreciate what you do. For instance, how do you deal with a marketing department that has the attitude, "We've sold this product, you design it," or with your fear that the marketer is misrepresenting what you are developing?

Or what if your manager is technically savvy and brought back a piece of IT equipment from a trip. It is acceptable to tell him, "I don't try to do your job; you shouldn't try to do mine." Our response to these and similar issues is careful education so that our colleagues see the error in trespassing on territory related to IT.

Attitude is very important. Be cooperative and courteous. You can do it right and keep people around or do it wrong. Any unpleasant situation can be made even less pleasant by a negative attitude.

This is about marketing ourselves, so here are some helpful hints:

1. Marketing is about educating. Educate people around you to help them understand how our jobs are interrelated. Cultivating relationships helps in achieving this. Reveal yourself in ways other than technical; hobbies and other general interests can help us connect to other people. In addition to books, mouse and toolkit, place pictures of families, pets, etc. in your workspace.
2. Let management understand what you do, that you are not just "staring at computers." Take the manager around; draw analogies. Send in periodic status reports of major accomplishments. Even if management does not demand this, it's a good idea.
3. Get your users adequately informed by warning them about changes. Do you let them know ahead of time when bringing the system down? Do you warn them about the database server? Send mass mailings; send enough, not too much. Package your regular suggestions for users in "Useful Tips." Give users a chance for feedback. It is legitimate to occasionally ask, "Is it useful?"
4. Create opportunities and look for the next problem to be solved. Look for opportunities to make yourself valuable. That's part of self-marketing! Find a niche for yourself. When people know who you are and what you do, they will come to you. How else can you keep your reputation as a miracle worker? Do something different: for instance, publish an article.
5. Keep a good problem-tracking system. Return phone calls and reply to emails. If a problem is not resolved quickly, acknowledge and give feedback. If you cannot get a problem solved, do not blame anyone; and if you don't know, say so. "I will research it" is a good response. Be honest about your capabilities.
6. Take a vacation. Here is a cool suggestion: have some toys on your desk that could relieve tensions or put them in a box with a label that reads, "Five-minute stress relief box. Feel free to use." When you are able to get away for vacation, put things in place that make the system work while you're gone. NO "I'm going for a week; let's see how they cope" attitude.

In closing, the panel members were in agreement on the following: "Make yourself a light. Be the illuminator" (Karen Ken); "Whatever you do, own it" (Dan Klein); "You make things, that's the goal. Work as a team with a sense of duty" (Strata Rose Chalup); "Be not ashamed, but be ye not arrogant either" (Karen Ken).

Sysadmin, Stories, and Signing: Learning from Communication Experts

David Blank-Edelman, Northeastern University

Summarized by Jim Hickstein

Sysadmins have to talk to each other, and to "other species," in other fields, especially when diagnosing system and user problems. The speaker brought perspectives from two other disciplines: storytelling and interpretation (specifically in American Sign Language).

Storytelling has a long tradition in the sysadmin community, but it has an academic underpinning that most sysadmins aren't aware of. Its mastery requires application, study, and practice. Yet in 20 minutes, the speaker gave a veritable short course in storytelling, which anyone would do well to take. He used the various methods along the way, repeating the part about repetition, using silence for effect in the part about silence. (The slides are good, but they don't do justice to this performance.)

Stories are good for sequential or related events; making diverse information coherent; passing on lessons (either overtly or implicitly). They fulfill social roles, as in establishing one's membership in a community. Stories make experience reproducible and reusable, and they do so safely (i.e., with a happy ending). Stories are good for constructing layers, which the listeners can then follow, especially in complex technical situations.

I won't try to reproduce the whole course here, but the lesson was clear: If you learn to tell stories better, you will be a more effective sysadmin.

He told a story about a difficult network problem escalating through a front-line

technician via online chat. It was an oft-repeated scene of a failure to communicate. But the technology was not really the problem: The parties seemed to speak different languages (though both spoke English) and had different backgrounds and mindsets.

What was needed? An interpreter!

The speaker then went on to talk about interpretation, in general as well as how it differs from translation. Interpretation is live, and the interpreter can't go over the "source text" more than once. Generally defined, interpretation creates in the mind of the "target" the same idea that exists in the head of the "source." It is subtle and difficult, especially when no direct translations exist: The interpreter must be able to move in two cultures and make the necessary mappings between them, accurately and in real time.

In ASL, for instance, pronouns are spatial: One doesn't say, "And then he said X, and she said Y." One creates people in space, in front of the speaker, (Z is here and T is over here), and then the "saying" happens in that particular place. Another one of the many challenging aspects is shown by the difference between "leave the party" and "leave the car at home." Polysemous words, density and context mismatches, preserving register...the list goes on. Affect and intent must be conveyed: The way something is said is very important to its meaning. It is what the listener hears that matters.

Mapping this to sysadmin communication doesn't take much imagination. The potential for misinterpretation is large; SA has many "rich points." One must use feedback to detect a snag, then go back and find the knot.

He finished with a taxonomy of useless support email requests, one of which read in its entirety: "Something is wrong and I have know idea what." (The speaker's first reaction, before seeing the subject line, "Printer help," was, "Yes, I have days like that myself, sometimes.")

Perl 6: The Science of Perl, aka Studies in the Ballistic Arts

Larry Wall, Creator of Perl

Summarized by Steve Wormley

Larry started off with a brief overview of where Perl came from. Perl has roots in linguistics, computer science, art, and common sense. In addition he discussed how Perl draws from ecology, math, and golf, among other things.

Perl was described as initially a way to combine the "manipulexity" of C and the "whipuptitude" of shell in one language. And Perl was designed to continue evolving into both. One major feature of Perl is that it is designed to hide the fancy stuff. In addition Perl behaves as a natural language. Some of these aspects of the language include: you can learn as you go, you can learn something once and use it many times, there are many acceptable levels of competence, and there are multiple ways to say something.

Another important part of Perl is the culture. The Perl culture, like some others, accepts newcomers, is okay with subtribes, encourages sharing, captures knowledge, encourages cooperation, and has fun. Perl 6 started by placing a request for comments for the new language. They received 361 comments. The Perl 6 team decided to take the Winnie-the-Pooh Approach: Think Things Through Slowly. They wanted to keep everything good and throw out everything bad. The final goals for Perl 6 were simplification, power, better OO programming, better functional programming, and better pattern matching.

Some of the new features and changes include: no more double parsing, comments work better in patterns, simpler precedence rules, removal of special variables, no more parentheses on conditionals (now whitespace dependent), and blocks are now closures. Full type signatures will exist, there is a new aliasing operator, and there will be vector operators.

One new aspect of Perl is that variables will have properties, such as a compile-time property of "constant" as well as runtime properties. These properties can also be accessed as methods. There are also new smart match and smart switch statements. Explicit exception handlers will now exist (try, catch, throw). New OO support will include opaque data which must be accessed by methods, and also the possibility for multimethod dispatch (the functions called depend on the types used).

The new pattern-matching support means that patterns are no longer interpolated as strings, the use of brackets is consistent, there are no postfix modifiers (all prefix or defaults). Other features will be new modifiers, meta-syntax, full grammar support, easy parse-tree generation, and grammar inheritance.

Finally, Larry mentioned that Perl 6 will be able to use Perl 5 modules. And to create Perl 6 code there will be a Perl 5 to Perl 6 translator.

How To Write a Book with Someone You Don't Know: Internet Collaboration for the Truly Geeky

Tom Limoncelli, Lumeta; Christine Hogan, Independent Consultant

Summarized by Kuzman Ganchev

Tom and Chris started the presentation by comparing the process of writing a book to that of managing a system administration project. To write their book, they used familiar tools such as SSH, CVS, and make, and had to deal with common system administration problems: security and data integrity. Their job was more difficult in that they lived two (and later, five) time zones apart and didn't know each other. The book, *The Practice of System and Network Administration*, is divided into four parts, 32 chapters, and three appendices. The presentation focused on how they had gone about writing their book.

First, they decided on a set of standards for formatting, tools they would use, and terminology (such as "customer" vs.

"user"). Then they used a top-down approach to plan out the rest of the book.

They divided the work by splitting up the chapters between them, and specified an explicit development cycle. They used their scarce meetings to do high-interaction brainstorming, used the phone for problem-solving sessions, and organized the logistics via email. They highly recommended automating as much as possible. For example, they used CVS to automate synchronization, Perl to generate the tables, and make for pretty much everything. They used open protocols, such as SSH and LaTeX, so that they could work from any platform.

The presentation ended with some comments about writing a book. They warned that the financial rewards are not likely to be great – minimum wage is above average – and that it takes a lot of work; they devoted two years of their lives to the task. Finally, they gave some advice for aspiring authors: Interview your publisher as you would an employer, negotiate hard on contracts, and retain a lawyer.

REFEREED PAPERS

SERVICE, RISK, AND SCALE

Summarized by Jim Hickstein

Application-Aware Management of Internet Data Center Software

Alain Mayer, CenterRun

The speaker described a new product that can help manage large groups of Web servers and their related applications. The product guides the user to "capture" the essence of an existing application (for instance IIS, all relevant content, ASPs, configuration files, etc.) from a "baseline" server into a central repository. Then it can be pushed onto new servers.

The master server contains the repository and certain engines, remote agents on baseline, and "managed" servers. Any server can be baseline and/or managed. The product encourages a workflow: deploy on baseline first, tune it using existing tools, then capture.

The system embeds version control. Replacing a managed server can be done in minutes. Objects inside the system include resources of various resource types, each type having a resource handler. The handlers are deployed on the remote agents to do the capturing (pulling) or management (pushing) of applications and state. The system is extensible by adding resource types and handlers.

The field is wide open for new research in modeling, config generation, rollback, policy-based management, among other areas.

Geographically Distributed System for Catastrophic Recovery

Kevin Adams, Naval Surface Warfare Center

The speaker described a disaster-recovery system that continuously copies data to a backup data center over a wide area network, at a steady rate that is just fast enough to meet the requirement to not lose more than N hours/days of data. Constant network utilization maximizes the cost-effectiveness of using a switched WAN rather than a private line. IP quality-of-service (QoS) guarantees adequate total throughput. The minimum and maximum data rates are nearly equal.

When you can eliminate all small "single" points of failure, the entire data center becomes the new single point. High-availability (HA) solutions like local, shared resources; disaster recovery (DR) wants things separated; HA eats bandwidth; DR wants distance – bandwidth is a problem.

They wanted to copy an HA system – migrate process, data, network identity, "heartbeat" – but tried to minimize the bandwidth required. A relatively low-bandwidth pipe would also minimize the impact on the primary site. A private circuit was considered, but distance and other factors argued for using a packet-switched WAN. In this case, they wanted to maximize the link utilization, for best cost-effectiveness.

The basis of the system is point-in-time imaging (snapshot). You dribble a copy of a recent snapshot to the remote site, constantly. The snapshot interval, and thus the data rate (given a fixed size) depends on the question, "How much (new) data can we afford to lose?" If it's 24 hours, that's the cycle; you need to copy N GB per day, depending on the size of the data set for each application.

The copy uses traffic-shaping to limit the transmission rate to a fixed upper bound, and IP QoS to guarantee minimum bandwidth equal to the maximum bit-rate, to ensure completion within the cycle.

Embracing and Extending Windows 2000

Jon Finke, Rensselaer Polytechnic Institute

The speaker described a meta-directory integration project that provides all students, faculty, and staff with a single username/password for all computer system access. Rather than modifying all authentication clients to use a central server, the username and chosen password are pushed out from a central system to several different client systems, including Active Directory (AD).

The institution needed AD for students, faculty, and staff; Exchange email for staff; and password and account synchronization across all platforms. Each person should have exactly one username/password. Certain Web services would tie into it. They also wanted this mechanism to manage email addresses, so the *user@rpi.edu* alias could be directed to any of numerous internal mail systems.

Existing administrative structures are not always along department or division lines. Some groups go their own way.

But DNS is centralized; no delegation, ever.

Currently, of 2000 employees total there are about 520 who use Exchange exclusively and 150 who are "casual" Exchange users. Windows 2000 authentication spans 400 public workstations and a number of administrative Web applications; a ticket system; and so on.

The speaker presented a graph of the distribution of mail systems by division: Exchange exclusively, other mail (department), central system (POP, *user@rpi.edu*). Administrative departments are mostly on Exchange; academic departments mostly not, yet.

He then showed a diagram explaining how systems are linked during a password change. The user interacts via HTTPS; that Web server encrypts the password in a public key and stores it in a change queue in a database. The encrypted password is shortly pulled from the queue, decrypted on the password-change server using the private key, and distributed to the several authentication systems, including NT domain servers. It does not happen instantly, but password propagation times were charted. Most were under 90 seconds.

PRACTICAL THEORY

Summarized by Kuzman Ganchev

Stem: The System Administration Enabler

Uri Guttman, Stem Systems

Uri presented Stem, a framework for creating tools to automate system administration. The Stem building block is called a cell. These are written in a custom-made declarative configuration language, and are executed by a runtime daemon called a hub. Uri presented a few example Stem programs. Of course, the first one was the obligatory "Hello world," which in this case conducts a conversation by replying with a greeting. He then went on to demonstrate more complex but trivially written examples, including a remote log-monitoring program. This appends data to a remote log and log file status changes (such as creation, deletion, and truncation) to a separate file.

Stem configuration files contain the definitions of cells, hubs, and portals through which hubs communicate with each other. Stem is implemented in Perl using an entirely peer-to-peer architecture, supports modules (that administrators can write to create more complex cells), and allows encryption using SSL.

Pan: A High-Level Configuration Language

Lionel Cons and Piotr Poznanski, CERN, European Organization for Nuclear Research

Lionel Cons started his presentation by introducing the Large Hadron Collider (LHC), what will become the world's largest particle accelerator, being built by the European Organization for Nuclear Research. This facility will produce enormous amounts of data. After on-site filtering, 10 petabytes will need to be stored to tape per year. The project will require 2 petabytes of disk storage, and over 100,000 processors. Pan is designed as part of an approach to solving the incredible system administration requirements of a cluster-computing project of that size.

After this introduction, Lionel presented some principles of the system administration project, such as automation, abstraction, and the use of configuration policies. The overall structure of the system would be a loop containing four components: the cluster, a monitoring database, an "operator," and a configuration database. The monitoring database collects information about the cluster, which is then examined by the "operator" – probably a combination of automated tasks and human administrators. This then modifies Pan source code, which is compiled into XML and stored in the configuration database, from where the clients retrieve it.

The language they required needed to be high level, be declarative, avoid duplication, support powerful validation and distributed administration, and be domain neutral.

Pan stores information in a tree structure and supports template manipulation and strong validation of data. It is licensed under the "European Data Grid License," an open license. It was designed to be portable but has not yet been ported beyond its original platform – Linux on i386. Finally, the project is in its early stages; Pan is not yet being used in production.

Why Order Matters: Turing Equivalence in Automated System Administration

Steve Traugott, TerraLuna; Lance Brown, National Institute of Environmental Health Sciences

Steve Traugott presented what he calls a "theory paper." He did not go into any formalism in the presentation but instead focused more on the paper's conclusions.

Traugott argues that in many production systems, there is a tendency for system administrators to make changes by hand instead of using automation tools. He calls the resulting system state "divergent," meaning that the difference between the baseline machine state and the current state is greater than expected, making rebuilds complicated, or requiring backups for the entire operating system. A different situation that he calls "convergent" involves an automated tool synchronizing changes between hosts (hence making them closer to each other). He claims that this is an ongoing procedure, since the multiple hosts are never quite identical. Finally, a "congruent" system is one where all the hosts start out identical and all changes are performed on them using a deterministic automated and repeatable process.

Traugott concludes that maintaining a congruent system is the least-cost

method to guarantee that a host can always be restored to its working state, especially if multiple identical hosts need to be kept (for example a Web-server farm). In particular in the long run it pays to reinstall systems from scratch to bring them to an identical state rather than to work with the disparate systems. Traugott recommends a tool he helped write called isconf to deterministically automate changes to different hosts.

LOGGING AND MONITORING

Summarized by James O'Kane

A New Architecture for Managing Log Data

Adam Sah, Addamark Technologies

When you have as much log data as Yahoo! does, you need new methods to store and query it. That's why Adam Sah presented a Log Management System (LMS) called Addamark. Some of the goals of this LMS were to handle 10's and sometimes 100's of GB of data per day, parse and query arbitrary log formats, be highly available and be able to keep the original files available in compressed format. Addamark achieves this by using a cluster of machines, and an extensible SQL-like query l anguage.

MieLog: Interactive Visual Log Browser for Inspecting Log Information

Tetsuji Takada and Hideki Koike, University of Electro-Communications

When you are looking through logfile after logfile, having a tool like MieLog, c an be helpful. MieLog, presented by Tetsuji Takada and Hideki Koike, gives an in teractive visual tool that can highlight key data. An administrator can see keyw ords, periods of high log activity, or high word frequencies. Everything is color-coded so you can see at a glance if there is a problem.

Process Monitor: Detecting Events That Didn't Happen

Jon Finke, Rensselaer Polytechnic Institute

But what happens when a service doesn't run, and therefore nothing is logged? Jon Finke addresses this problem with a tool called Simon. With Simon, services l og when they last run into a database, and when a service has not checked in wit hin its configured window, a notification is sent to the administrators. For exa mple, if a service should run every 24 hours and the last time it reported in wa s more than 25 hours ago, notification is sent.

SERVICE AND NETWORK UPGRADES

Summarized by Jim Hickstein

Defining and Monitoring Service Level Agreements for Dynamic e-Business

Alexander Keller and Heiko Ludwig, IBM T.J. Watson Research Center

Alexander Keller outlined a software system that manages service level agreements, defined in such detail that a computer can automatically evaluate needs against offered services, and actual performance against guarantees, all in aid of permitting e-business suppliers and consumers to find each other dynamically. Dynamic e-business is created and dissolved on demand. For instance, a Web site's inventory, cart, and payment services might be distributed among several providers. With a dynamic system, the Web server could select, for example, payment providers on demand, based on a cost bid.

What do SLAs have to do with the daily chores of the sysadmin? In fact the sysadmin is constantly making such evaluations: What is the cost to guarantee a response time of less than one second? How much should we bill a customer for throughput of 1000 transactions per second? How much revenue is lost per hour of downtime? Can you accommodate another customer and more workload? How would this impact SLAs with other customers? SAs will become involved in this, because they have the knowledge underlying it. The speaker outlined the structure of the system, composed of SLA parameters, metrics, and functions. Some are resource metrics, others composite metrics; for instance, a function might define the peak value of a metric over a given time period. Various services (measurement, evaluation) might be delegated to third parties. The specification is flexible, using a formal language. The software package, WSTK 3.2 with SLA-compliance monitor, can be downloaded.

HotSwap – Transparent Server Fail-Over for Linux

Noel Burton-Krahn, HotSwap Network Solutions

Several techniques exist for adding fail-over capability to certain parts of a computer system, with certain limitations. But most of them don't address the problem of a failing server which has a live application state and, especially, open, long-lived TCP connections.

The speaker presented a solution for transparent fail-over of Linux servers, which preserves internal state and connections. It does this by running entire virtual servers on separate hosts, sharing a virtual IP address, synchronized in near real time over a local network. A diagram showed a typical high-availability Web application, with network load balancers distributing HTTP requests to Web servers, and these talking to a back-end database. The HTTP connections are quickly over, but the database connections tend to be long-lived, and the database server itself becomes a single point of failure. Commercial database solutions exist to make this part fault-tolerant, but they tend to be expensive, and even the front ends will occasionally show a failure to a user, when, for example, a Web browser times out. A fail-over system should never lose data; the clients should never be aware of a failure; no connections should be broken; the cost

should be low; and it should avoid forcing a rewrite of existing server processes.

Naturally, there are trade-offs. Cheap backups mean long recovery times, whereas full replication and quick recovery is expensive. The goal of this system is to replicate a server on another box, without a rewrite. It replicates the network, TCP, and internal program state, even memory, by knowing and duplicating all external stimuli coming in through trappable system calls. This assumes the server processes are deterministic, which is often true, though OpenSSL had trouble until an uninitialized memory bug was fixed. Timing-related code and direct hardware access may also break this assumption. Performance may be an issue, of course, and the network traffic between master and slave may be large. But tests so far show a reasonably good result. The author's Master's thesis was a demonstration of the system serving streaming video. On HTTPS downloads, there was about 9% degradation compared to a single server.

Over-Zealous Security Administrators Are Breaking the Internet

Richard van den Berg, Trust Factory; Phil Dibowitz, University of Southern California

Path MTU Discovery (PMTUD) is used by many TCP implementations, usually to good effect. But a growing number of sites on the Internet have overly restrictive firewall rules that block certain critical ICMP packets, resulting in whole classes of users who simply cannot see these sites. They create self-inflicted PMTUD "black holes." More than a few are security-related sites run by people who ought to know better. The authors are calling for better education on this issue and running a Web service that users can check to see if a given site is in a known black hole. Certain ICMP packets have been an avenue for some attacks, so security administrators tend to decide that all ICMP packets are dangerous and none strictly necessary. They are wrong about that: ICMP is not an optional extra. It is an essential part of IP, and filtering it out entirely will break an IP network. Some ICMP types are just more important than others. The MTU (maximum transmission unit) is the longest IP packet that will cross a given network link. For best bulk-transfer performance, two IP hosts should send each other packets that are as large as possible for the end-to-end network, but no larger. IP can fragment packets, if they exceed the local MTU at any point along the path. But the sender can set a bit, called "don't fragment" (DF), to say that a packet needing fragmentation should instead be dropped, and an error returned to the sender. This error is ICMP type 3 (unreachable) code 4 (fragmentation needed and DF set).

Path MTU Discovery works by setting DF on all the packets in a connection; if the ICMP error comes back, the MTU is reduced and a shorter packet sent. If no error comes back, the sender assumes the path MTU was large enough to accommodate packets of this size, and it proceeds. If a firewall blocks all ICMP packets returning to the sending host, such a connection will not work: The sender will time out and re-send the same, too large packet, and eventually give up. For a Web site, the user sees the connection established, but nothing ever comes out. The users most affected are those with a slightly constricted MTU, typically because their Internet connection requires a tunneling method such as GRE, PPTP, or PPPoE. Many consumer-broadband users are in this group. Their number is growing quickly. The authors have started the MSS (maximum segment size) Initiative, to try to educate those responsible for creating PMTUD black holes and to offer help in fixing them. They also list their successes and failures.

NETWORKING TRACK

Large-Scale 802.11

Tim Pozar, Late Night Software

Summarized by David Berg

Tim actually titled his presentation "Long Distance Wireless Networking Using Non-Licensed Radios." This session presented a top-down view of wireless networking on a scale larger than your average single-access-point LAN. Tim began with an overview of topologies, applications, and pros and cons of 802.11.

After laying out the basics, he moved into a more practical arena. Discussing the design of networks, he mentioned both "site surveying" and "engineering the link." "Engineering the link" covered signal loss/gain and attenuation – topics that segued nicely into his comments on hardware. Tim presented various examples, including pictures, of classes of antenna and access points. One of the more fascinating access points was the home-brew model, for which he, unfortunately, didn't provide instructions.

Pozar continued his speech with several brief remarks on security, including the forthcoming 802.11i standard. He concluded with a round-up of what we can look forward to in the 802.11 family and a list of books and Web sites of particular interest to the aspiring large-scale wireless guru.

SECURITY TRACK

Internet Security: Beyond Firewalls, Passwords, and Crypto

Peter H. Salus, Matrix NetSystems

Summarized by David Berg

Salus' presentation clued in the audience on the myriad threats that the Internet faces and that lie beyond the control of any local administrator. Peter presented the information using the analogy of a medieval fortress under siege and a wealth of graphs depicting reachablility and packet loss on the entire Internet. The discussion started with several slides on the history of worldwide Internet growth and the general state of the Internet at present.

Emphasizing the siege theme, Peter proceeded to demonstrate the effect of some of the recent viruses (April Fool's Virus) and DDoS attacks on the overall

flow of traffic across the matrix. He continued with other, perhaps less obvious, threats to IP traffic, including the severing of one of the oceanic fiber lines connecting China to the world, the 9/11 terror attack, and the bankruptcy of WorldCom. He finished the slides with the October 3, 2002, DDoS attack in which the 13 root DNS servers were attacked.

The session ended with a discussion with the audience on the possible solutions to these types of disruptions. Peter suggested that DDoS attacks might one day be prevented with an IP "early warning system." Until that day, as Peter's answer to one participant's query highlighted, the solution is active monitoring.

The Promise of Privacy

Len Sassaman, Consultant

Summarized by Martin Krafft

Len Sassaman has been involved with PGP from the early days, which puts him in a role to analyze the position of PGP and its relatives today. To sum up his talk, everyone is screaming for privacy, and yet nobody uses the tools available. Topics ranged from basic crypto to why PGP and similar products are failing.

Privacy comes in various forms: financial privacy, communication privacy, privacy of stored data. The need for privacy in all these areas is high. Modern technology poses new risks in the form of credit card fraud, ID theft, and general trust in the law to protect oneself. One of the answers to the general problem of protecting privacy is cryptography, which has seen great successes. PGP ("Pretty Good Privacy") was released in 1991, and technologies like SSL/TLS, S-MIME, and anonymizers are also still in widespread use. Consumers understand the threats, and the technologies are there, but privacy aspects of the current Internet are frightening.

The problems that Len isolates touch on almost every aspect of cryptography and related technologies. From user apathy to developer incompetence, from politically influenced decisions to the intractable problem of usability, cryptography is experiencing a number of problems as it tries to be accepted into everyday use. As an important point to back up his arguments, Len mentioned various fields in which cryptography has improved: where the user interface is simple, where there is a real need, and, last but not least, where it's actually used.

But cryptography is suffering from the problem of weak links in a chain. Unless everybody uses it, it is not going to be useful on a broad scale. A vast number of people don't use cryptography because it's not standardized, not readily available, or simply too confusing. Len questions whether it would help if the entire theory around encryption could be reduced to processes similar to sealing a letter and sending it off. He points to various attempts at making crypto easier, including PGP and TLS, as well as more high-level services like Hushmail, Zendit, and Lokmail. Most of these try to reduce the user interface to the bare minimum, with TLS being "the best" because it is opportunistic and invisible.

In conclusion, Len wants to see the technology simplified for the user. He wants friendly user interfaces, better integration, no room for individual error, and everything to be open-hooded. He wants cryptography as a standard, with the proper usage being the only usage. You are not alone, Len. Who's going to do something about it?

My Years with the NSA Red Team

Tim Nagle, TRW Systems

Summarized by Robert Beverly

Nagle spoke to a capacity crowd, underscoring the interest people have in one of the government's most secret organizations. The NSA Red Team is a group of specialized individuals whose charter is to protect information security, including voice, data, and encryption. Typically, they attempt to compromise the security of a network and the hosts within the network. Contrary to prevailing opinion, the NSA only offers this service (popularly referred to as "Red Teaming" systems) to US government networks and the networks of government contractors. Further, the NSA will probe only with the explicit request of the organization. In some cases, parts of the network that were considered critical were off-limits to the Red Team. For example, the air traffic communications at an air base were not probed. A number of people were skeptical, believing that this prepared the organization, in effect, for a not very rigorous NSA "attack." Nagle emphasized that they were working together with the organization they probed, not against them. The team never tried to exploit social engineering to compromise systems.

The NSA Red Team grew out of the 1987 Computer Security Act that divided the responsibility between NIST and the NSA. "Eligible Reviewer" was the code name for the summer 1987 DoD exercise to improve the war-readiness of government computer systems. The exercise evaluated vulnerabilities of the systems and scripted out what might have happened in the event of a malicious compromise. For instance, sending troops to the wrong location, disrupting supply chains, and so on.

Despite the prior warnings, the Red Team invariably found security holes. In order to prove compromise, the team generally left a file or some other evidence of the security hole. Every keystroke was logged to aid forensics and reproducibility. Often, the team was required to prove what they did and did not do.

Much to the chagrin of at least a few members of the audience, the talk did not discuss any technical specifics of how the Red Team compromised networks. However, Nagle provided interesting insight into the policies and procedures the NSA follows.

The Intrusion Detection Timeline

Paul Proctor, Network Flight Recorder

Summarized by Abiodun A. Alao

We are slammed on all sides – viruses, rogue insiders, employee error, software bugs, corporate spies, Web defacements, script kiddies, password crackers, network vulnerability, worms, Trojans – the list seems endless. The economic impact of malicious codes has grown exponentially to over $13 billion a year.

The number of attacks in the first three quarters of 2001 rose by over 60% compared with the entire year 2000, representing a loss of almost $380 million by corporations, government agencies, financial institutions, medical institutions and universities! And it's going to get much much worse.

The paper focuses on knowledge for selecting and employing information security technologies that are appropriate, meet organizational needs, are able to contain known risks and stated requirements, and pass a cost-benefit analysis.

Most intrusions are the result of known vulnerabilities or configuration errors where countermeasures are available; 99% of intrusions could have been prevented with patches, updated servers, etc. A direct reaction to vulnerability would be to close the window to exposures, but it is important to identify all such windows as they emerge. Making everything secured stops business and drives administrative costs through the roof. This returns us to the issues of the cost-benefit analysis of available solutions. For instance some threats may not materialize or their effects may be muffled and not as significant as anticipated. Investing huge sums to prevent such attacks may not be economical or efficient.

How then can you defend your organization? There are six major steps:

1. Analyze risk and classify resources: You have to set your enterprise-specific requirements. Identify all the source of risks and their costs in terms of potential damage to systems, loss of opportunities to do business, etc. Also estimate the value of each resource and the implications of the breach of any of them for the organization. The most critical and vital resources should get the best protection. Some questions to consider include:
   - What threats are most relevant to your business?
   - How critical is the data?
   - Where does it reside? What is its value?
   - How do you define an attack?
   - What are the technology value propositions?
2. Anticipate: It is important to anticipate potential problems by creating effective policies in the areas of security, auditing, configuration, detection, access, boundary, and application design.
3. Protect: Protect computers to reduce the threat of compromise from the inside or outside. The strength of a network or system is determined by its weakest link. Therefore, ensuring adequate protection of *all* systems on the network is essential. The following specific steps should be taken: assess computers for vulnerabilities; install latest patches regularly; use best industry practice; keep anti-virus software updated; disable Java, JavaScript in browsers; turn off macros in applications; and back up servers and workstations.
4. Detect: Prevent attacks that are known, detect attackers probing for weakness, and direct attackers into honeypots. This will make hacking more difficult and less rewarding, and may reduce the incidence of attacks. Detect network probes as attackers search for vulnerability to exploit network scans, port scans, and systematic activities. This is usually accomplished with IDS technologies for log analysis.
5. Respond: Well into the attack or shortly after an attack, forensics, and correlation will help determine what has happened or what is currently happening. Response must be timely and appropriate; that is more than enough to solve the problem and deter further attacks.

Check constantly the integrity of all files and fix problems as soon as they are detected to minimize the cost of such attacks. Review logs to reveal patterns of likely attacks. Gather evidence and apply trending and long-term analysis to determine further activity. This makes it possible for firms to anticipate attacks. Proactive firms are able to beat the attackers. Finally, it is important to report and log all attacks and attempted attacks to ensure that the organization has in place adequate data to plan with and to use for prevention.

Various technologies were examined, including system call trapping technology (Intercept, OKENA, Trojan Trap), honeypots/decoy technologies, network IDS, HIDS-Log analysis, and file integrity checkers.

Security is a process, not a destination; use the right technology for the right problem.

Slides and other security resources are available at *http://www.practicalsecurity.com*.

GURU SESSIONS

NAS: Network Attached Storage

W. Curtis Preston, The Storage Group

Summarized by Kuzman Ganchev

When I came in, the discussion had already started, and NetAppliance filers were being discussed. Essentially, the problem with these is that you have to keep the NetApp filer around as long as you want your data. Curtis gave a few examples (without names) of companies who still have to keep around archaic technology, because it's the only thing that will read their old backups, which they still use from time to time.

The discussion then moved to non-tape storage. Curtis mentioned a service at e-vault.com, for backing up a small amount of data over the Internet; this is probably best for personal data – configuration files and other compressible information. For a small office, disk-based backups can be a better solution than traditional tape. Curtis cited a backup failure of up to 40% in small office environments, because of failure to insert the next day's tape after a tape is ejected. Though taking media off-site for disaster recovery is not possible with a disk-based solution, at least the data is being backed up.

Alacritus Software, a Livermore-based company writes storage software that enables a disk-based system to act as one or more virtual tape libraries. They do not actually provide out-of-the-box solutions directly but have partnerships with third-party vendors to do so. Curtis suggested backing up to a disk-based device and then periodically duplicating those to actual tape to be taken off-site or stored in archives. This is better than backing up the device to tape, since a restore from tape only requires one operation, as opposed to two in the case of backing up the backup device.

Discussion then moved to the Quantum DX30, which are disk arrays used as backups for quick restore. According to Curtis, these are not quite as small as originally intended due to unresolved cooling issues.

Perl/Scripting Gurus

Daniel V. Klein, LoneWolf Systems; Mark-Jason Dominus, Plover Systems

Summarized by Abiodun A. Alao

Larry Wall once said, "Most of the programming out there is not done by Perl experts...they learn by experience to do better over time, and eventually they become experts." We took a step in that direction with Perl scripting gurus Dan Klein and Mark-Jason Dominus. Just type these guys' names into Google and you'll get more than you ever need to know about Perl. Doesn't get any better than that does it? Ah, but it did. A few unannounced guest gurus showed up: Matthew Barr and Larry Wall. Where else but at LISA?

We were treated to a guided tour through the coding of Larry's own home automation and monitoring setup, accompanied by many fascinating side trips into his life and personal interests: X10 problems, techniques, war stories, human-readable code. Are we asleep or awake when the thing goes bump in the night? It does make a difference, at least in Larry's home. I pity the poor mice. He didn't touch on mousetraps; perhaps that's a question for next year.

Dan Klein could not be outdone, leading to displays of several such systems. Water-flow monitoring, and why you should care. A graphical display of furnace operation related to temperature inside and outside the home. At the cabin on the lake, the water temperature at the surface and underwater. Ways not to waterproof a temperature sensor, complete with graphic descriptions of failure modes, and at least one method that works.

Mark-Jason Dominus happened to mention his Perl quiz of the week. Check it out at *http://perl.plover.com/qotw/*. A new quiz every week followed later by sample solutions. No better way to learn (except to get paid for it). This week's entry: Find all the anagrams in a list of words. And, they were off! Amazing how much can be done with a single line of Perl. Oh, forgot to mention they were to be sorted alphabetically . . . no problem. Oh, and if there are more then two words . . . and this isn't even the "expert" quiz.

Some tidbits we grabbed out of the air:

> "You can deal with unreliability in automation . . . a little bit."
>
> "How does one become a Perl guru? Volunteer a lot, try hard things, fail a lot, and learn."
>
> "Tired of chomping and putting \n at the end of every print statement? Try perl -l."

How do you top all that? Perhaps a Perl script to generate unique pattern sets for a quilt, then having your wife sew it. And convincing her it's a gift! Hmmm, don't try that at home. I guess that's why these guys are the gurus.

Email/MTAs

Eric Allman, Sendmail

Summarized by Martin Krafft

This year's guru session on MTAs and email was well attended, led by Eric Allman, author of the infamous Sendmail and current CTO of Sendmail, Inc. It wasn't a big surprise that the first questions were about spam. Eric talked about the simple anti-spam methods in Sendmail (which are still more advanced than most other MTAs), like per-host connection throttling, tweaking rule sets, milters (mail filters) and RBL, and he referenced Spamassassin. The next question concerned remaking SMTP, cleaning up its fundamental flaws as part of the anti-spam war. Eric agreed, but he stressed the extensibility of SMTP and argued against a new protocol on a new port – port 25 is the mail standard, he argued, and changing standards is near impossible: If you splinter the Net by trying to introduce a new standard, you not only create chaos for email for some period, but you also make it possible for a company that would prefer that the Net run on their proprietary standards to get a foothold. He also addressed the single fax machine problem – either everyone employs the "new SMTP" or it is as useless as a single fax machine. The next question on spam dealt with a buffer/moderation queue in Sendmail, which would allow a postmaster to intervene in case of a spam flood. Finally, a couple of technical questions about MTAs and the RFCs yielded closer inspection of RFCs 2821 and 2142 about the type of email addresses one must accept. Even though <abuse> and

<postmaster> are listed in 2142, nobody really forces users to implement them. The empty address (<>) is accepted nearly everywhere, though. Rfc-ignorant.org was mentioned.

The discussion moved to the roles of SMTP and instant messaging. Eric doesn't seem to see their technologies fusing in the future, but he recognizes that users perceive them more and more as one and the same. Eric wished that he had actually implemented SEND, SAML, and SOML (which are forms of instant messaging) in Sendmail because this would have possibly standardized IM systems from the start. The audience noted that jabber (one of the later and more successful open source IM systems) is starting to implement queueing, so maybe the technologies aren't too far apart after all.

Eric then talked a bit about the forthcoming version (8.13) of Sendmail. Among many new features, it will include LDAP support and milters per socket, but it won't interface with Berkeley DB 4.1 (even though that's being worked on with the Berkeley folks). 8.13 still has some problems with the latest Linux implementation of flock(), which doesn't behave as expected. Eric announced the "Bat Book" (O'Reilly's Sendmail book) on version 8.12 for the end of the year and said he will release 8.13 before 2003 only over his dead body – he wants the book to be current for at least a while.

Performance and scaling comparisons between various MTAs came up next. Oracle's new mail product (which uses Sendmail) is not their first attempt at this market, but previous attempts were not commercial successes, which Eric attributes to the inadequate speed of the Oracle back end for a real-time mailer application. Performance comparisons between the big UNIX mailers Sendmail, postfix, and qmail cannot really be instituted. Eric believes that qmail does way too much sync-I/O. Following up on performance and I/O, a postfix admin asked if Sendmail suffered from the same problem as postfix when it came to journaling file systems. Eric carefully tried to answer by saying that Sendmail has had good luck with journaling file systems in the past. He does not know of serious implications or dangers when running the spool on a JFS. People also asked about a mail queue residing on a solid-state disk, with which Eric has had some success. Nevertheless, he suggests not putting the entire queue on it, just the hotspots.

The last set of questions was about queue consistency and lifetime, and the ability to back up and restore the queue on a live system. While other mailers have various kinds of problems with manual intervention of the queue, Eric notes that Sendmail's queueing strategy has been reworked to avoid collision for 60 years, so even injection of restored data into a live system would not mess up the consistency of the queue. However, Eric specifically does not recommend this on production systems.

Performance Tuning

Jeff Allen, Tellme Networks

Led by Jeff Allen, author of the Cricket SNMP monitoring tool, the performance tuning session was loosely organized and consisted of specific questions as well as general problem-solving methodologies.

Allen emphasized that one should always understand data and statistics in context. As an example, WebTV engineers could not immediately offer an explanation for a drastic dip in network usage for a particular day until they discovered it coincided with the broadcast of the Super Bowl. In general, one should always form a scientific hypothesis and test that hypothesis. When analyzing statistics, averages are generally of little use since the most interesting events (and those that cause issues) are outliers. Many distributions have strong modalities or heavy tails that negate the conclusions pure averages may find. Instead, use box plots, which display the mean, minimum, maximum, and quartiles for a given data set. Box plots graphically provide much more information about the data and reveal any hidden peculiarities.

Questions focused on the typical culprits of resource contention: network interfaces and hard disks. The discussion turned to Gigabit Ethernet interfaces on Sun equipment, where the performance was sub-optimal. Many factors may contribute to this, including the packet-size distribution and various TCP parameters. Allen explained the notion of the bandwidth delay product, the ideal number of unacknowledged packets in flight. The TCP window size provides receiver-initiated congestion control. To achieve maximum link utilization, the window size must be large enough to accommodate the bandwidth delay product. Sun also has the notion of TCP high-water marks, which control the rate in which user space applications may access kernel network resources.

Finally, questions about disk performance surfaced. First, one should determine whether the problem is in fact due to an I/O-bound device. The iostat command is ideal to observe disk and controller performance. If the disks are in fact the bottleneck, data should be stripped across as many disks as necessary (often five or more). In this manner, a single datafile is divided so that a piece of the file exists on each disk. Because disk read performance is the limiting factor, each disk can now read their portion of the file in parallel and fully utilize the controller bandwidth. Even though this will waste disk space, disks are relatively inexpensive today and this technique will yield much higher performance.

WORKSHOP SERIES

SYSTEM CONFIGURATION WORKSHOP

Summarized by Will Partain and Paul Anderson

The system configuration workshop, with 22 participants herded by Paul Anderson (University of Edinburgh), built upon the cfengine workshop at LISA 2001 (*http://www.cfengine.org/Workshop/*) and the Large-Scale System Configuration workshop in Scotland (*http://homepages.inf.ed.ac.uk/dcspaul/publications/wshop/*).

Anderson led off with an introduction to system configuration: Given a large computing infrastructure (dozens to thousands of hosts), how can we describe its desired state in a humanly tractable form? How can tools (better) use such a description to control the infrastructure?

System configuration tasks span an infrastructure's whole life, including pre-installation (e.g., BIOS configuration), operating system and software install, configuration of that software, managing changes to the infrastructure over time, and taking in feedback information about the infrastructure and recovering from faults.

Problems that arise in the design of system configuration tools include handling scale, diversity, and/or change; supporting modular management so that different people can control individual aspects of an infrastructure separately; providing an explicit representation of components (separate from the components themselves); providing higher-level views of an infrastructure (e.g., viewing a cluster as a single entity); making possible the desired level of consistency across systems; and security (of course).

Solutions to these problems have to choose between static vs. dynamic configuration (e.g., JumpStart vs. cfengine); getting to a "good" state by cloning vs. by scripting; declarative vs. procedural language (more below); centralized vs. distributed control; and synchronous vs. asynchronous operation.

Though the rest of the workshop talks described particular system configuration tools, the purpose of this workshop was to study tool-independent configuration principles.

Much discussion arose from the notion (raised by Luke Kanies) that existing tools comprise an unholy *mix* of model, language (to express an instance of the model), and implementation (of the language). We will do better when these concerns can be understood independently. Points raised in this session included:

- Ideally, the language should express *what* is true in a model of a configuration (a declarative approach), not *how* to make it true (a procedural approach).
- A model should be able to represent inter-machine relationships and be independent of implementation details.
- The model needs to represent dependencies between components, including runtime temporal dependencies. ("Service X must be started before client Y tries to use it.") Temporal constraints are even more fun. ("Kernel upgrades can only be deployed across lab machines on Saturday nights, except in exam week.")
- The "truths" expressed in a model and the "truth-checking" of a monitoring system need to be closely coupled (more below).
- The model and language must support devolved management. If more than one person is specifying configuration details, how do we know the total infrastructure still "makes sense"?
- The conversation continued about the importance (or not) of "ordering" in a model; see the Traugott/Brown LISA paper for one side of the story.
- There was some discussion of whether or not a good model needs practical backing by a CPAN-like Infrastructure Framework Library (suggested by Mark Roth).

A surprising issue that emerged in the discussions was usability: System configuration tools often fail to make headway because they are too hard to use. Possible reasons for this:

- Configuration tools are complex, with a steep learning curve, especially for small sites. Better user interfaces are needed (for both GUI tools and languages).
- A tool embeds its author's notion of sysadmin policy, which proves inappropriate at any other site.
- A configuration tool is most useful when it has complete control of the system. This is a big culture change for many sysadmins.
- Existing tools are a diverse, fragmentary bunch, each one covering just a part of the problem; learning enough tools to do the whole job is a daunting task.

An idea that gained immediate acceptance by the group was that there must be a strong connection between configuration, testing, and monitoring. Specific points raised:

- Monitoring and feedback of the *actual state* are crucial. "We have to embrace failure." (Andrew Hume)
- What do we mean by "testing" a configuration? How can we test configurations before deploying them?

Our sketch of this workshop should make clear that system configuration is an intellectual and practical challenge. The conversation will continue at LISA 2003 – interested configurationists take note! Until then, details about a configuration mailing list are at *http://homepages.inf.ed.ac.uk/group/lssconf/config2002/*, along with all of the materials (e.g., slides) from this workshop.

Education and Book of Knowledge Combined Workshop

Coordinators: Geoff Halprin, Rob Kolstad, SAGE; John Sechrest

Summarized by Rob Kolstad

This year, the Education and Book of Knowledge (aka sysadmin taxonomy) groups merged their workshops in order to learn each other's working style and interests. About 18 people attended, including organizers and individual contributors from both groups. The Education group included several people who were trying to run 10- to 18-week courses and a fellow from NYU who is implementing a five-semester Master's degree course in system administration (!).

Geoff Halprin presented his brilliant BoK history and motivation. We're all solving similar problems, and we need to work and develop from the same foundation. Let's address areas of personal ygrowth, organizational maturity, and a framework upon which we can capture "best practices."

Sysadmin is about "intricacy," the interplay of components that come into play when dealing with complex environments and a continuous stream of microscopic changes. Thus, there is no such thing as "a best practice." There are a *number of best practices* that we can capture.

System administrators ensure integrity of computing systems and assist users in maximizing effectiveness of their computing environment. System administrator roles include: troubleshooter, walking encyclopedia/user manual, toolsmith, researcher and student, tech writer, both strategist and tactician (today and in the future), doctor, and counselor.

Administrator tasks, challenges, and difficulty combine with availability and hidden costs to present issues with many details. Professional development proceeds along half-a-dozen paths. Standardization is needed since people change jobs frequently (every 1.5 to 3 years), and it takes six months to adapt to a new job. We must understand the nature of a problem space by breaking the problem into its components and understanding them.

Geoff covered several related programs and listed unique features of sysadmin. He also discussed professional development and "key areas of responsibility."

The BoK seeks to define a sysadmin and development maturity model. Several examples were given. The BoK is a reference framework that supports best practices, enables effective training, and feeds certification, education, and job descriptions by listing the core skills, knowledge, and disciplines of the profession.

Rob Kolstad echoed many of these same sentiments and discussed the over 2000 elements now present in the current BoK matrix (of tasks/knowledge and the various factors that affect those tasks). Creating the document is the next step, given this list.

John Sechrest talked about the Education Committee's work. "Last year, we had a workshop, and I asked a lot of demographic questions in an effort to learn how best to serve people and enable sharing of teaching ideas." He showed his goals and discussed accreditation at his university. He gave a list of about a dozen topics and sub-topics.

Hours of lively discussion ensued, with lots of time spent delving into topics like risk assessment and change management. Teaching techniques and paradigms were discussed, including the creation of virtual laboratories. A curriculum discussion group was formed for the purposes of creating a four-year curriculum (from whence other curricula will evolve). Attendees rated the day a general "thumbs up."

AFS Workshop

Coordinators: Esther Filderman, Pittsburgh Supercomputing Center; Derrick Brashear, Carnegie Mellon University

Summarized by Garry Zacheiss

The workshop began with status reports from representatives of both Arla and OpenAFS. The current released version of Arla is 0.35.10, which supports all *BSD UNIX variants, including MacOS and Linux. An 0.36 release is expected to branch before the end of the year. This release will include Themis, their `package` utility replacement, which includes features and extensions not found in the traditional AFS `package` utility. Themis should be a drop-in replacement for `package` . Improvements in Arla 0.36 include support for incremental open and support for UUID-based callbacks (via the WhoAreYou RPC). Additionally, the afs3-callback port used by Arla will change from `7111/udp` to `7001/udp`, and XFS will be renamed to NNPFS. Windows support will also be present in this release, along with a GUI ACL manager for MacOS X that integrates with the Finder. The MacOS X ACL manager will also work with the OpenAFS MacOS X client. Future goals include implementation of a cleaner and faster kernel/userland interface, and the addition of IPv6 support for AFS. Work on integrating Kerberos 5 and GSSAPI into Rx continues.

OpenAFS recently celebrated its two-year anniversary. Recent progress in OpenAFS includes the addition of fakestat; with this feature enabled, the AFS client will provide stat information for volume mountpoints not yet traversed without contacting remote file servers. This allows the use of graphical file managers to browse `/afs` without causing excessive hangs and timeouts. This feature is present in OpenAFS 1.2.7; OpenAFS 1.2.8 will include a further refinement to only present this behavior for mountpoints to volumes in foreign cells. Other recent features include ports

to MacOS X 10.2 and an experimental port to FreeBSD, further Linux client tuning, and modifications to the file server to use Rx pings to determine if clients are reachable before allocating threads to them; this prevents asymmetric clients from consuming all available file-server threads. Issues that OpenAFS is currently facing include recent RedHat Linux kernels (which break the OpenAFS client by no longer exporting the symbol sys_call_table), the minimal RedHat AFS client, and a forthcoming HP-UX 11 port. rxkad 2b, which will add Kerberos 5 support to Rx while still using fcrypt for encryption, will appear in a future OpenAFS release, most likely OpenAFS 1.2.8.

Other discussions included:

- Porting OpenAFS to HP-UX for the Itanium and to AIX 5.1 and later
- CERT's transition from Transarc AFS to OpenAFS/Kerberos 5
- Using AFS through a firewall
- Using AFS with Kerberos 5 and a Kerberos migration kit
- Performance benchmarks and tuning
- Common user errors – Backing up AFS cells
- AFS on MacOS X 10.2
- IBM's end-of-life announcement for their AFS implementation

The workshop closed with a roundtable discussion on what AFS needs to do to gain more market share. Support for files larger than 2GB, byte-range file locking, better support for Windows clients, and more training opportunities and documentation were all cited as being desirable for AFS to gain additional market share.

Advanced Topics Workshop

Coordinators: Adam Moskowitz, Consultant; Rob Kolstad, SAGE

Summarized by Josh Simon (with help from Rob Kolstad)

The Advanced Topics workshop was once again ably hosted by Adam Moskowitz. We first discussed what percentage of our time is spent on reactive versus proactive tasks, which varied relative to how close to the end user or customer our roles were.

We next talked about the various barriers to fixing problems, including technical ones, economic problems, problems of management not understanding, and so on. Many of these issues are discussed in the forthcoming SAGE *Short Topics* booklet on budgeting.

Our next discussion was on why we reinvent the wheel by recreating the tools for the same task again and again. Reasons include ignorance of preexisting tools, political factors influencing the decision (the "not invented here" syndrome), taste, where the tool falls in the issue of specific vs. general, and changing needs.

After our discussions, we went around the room to list our favorite URLs that might be unusual as information and humor. We went through system administration aphorisms — pithy sayings such as "Never send email in anger." SAGE will be making a poster of these. (Send your favorites to *kolstad@sage.org*.) Finally, we talked about things we learned in the past year and, as usual, made our annual predictions.

WORK-IN-PROGRESS REPORTS

Summarized by Peg Schafer

The LISA '02 WIP session went very well. We had some interesting submissions! Amr Awadallah created a lot of excitement with his vMatrix presentation. However, the crowd gave the LISA '02 WIP Whip to Jeremy Mates for his "Improving Productivity" (by reading your daily cartoons) presentation.

In order of presentation here are the submitters' own descriptions.

When the Trouble Is People, Not Technology

Chuck Pervo

cpervo@jonesday.com

Sysadmins of the world unite! Are you tired of being stepped on by others? Have you ever been in a situation where there was a serious problem and you were out-shouted in the problem resolution process by an unknowledgeable person? Or when the process was directed by politics rather than solutions based on causality, data, or reason? If the answer to these or similar questions is YES, you are not alone! Alva Couch has encouraged me to do a paper on this topic, including case studies and a manual on formal problem resolution practices, which will include a Robert's Rules-style set of guidelines that should preempt such time-wasting, stressful activity.

Tivo & MacOS X

Matthew Barr

mbarr@mbarr.net

After being encouraged by some seemingly nameless party, I've been conned into doing this. So, you get to hear about it. This WIP will focus on a MacOS X machine being the recipient of a copy of data from a Tivo. It includes information on connecting a Tivo to a TCP/IP network, enabling external control of the Tivo via HTTP and Web browser, as well as how the heck to export data from a Tivo to a Mac/UNIX system. I am also involved with a collaborator (who just happens to work at Apple :) on designing a GUI system for all of this.

The vMatrix

Amr A. Awadallah

aaa@cs.stanford.edu

The vMatrix is a network of virtual machine monitors allowing for fluid server mobility between real machines. By building the servers inside of virtual

machines, we can easily move them around. The applications that we are targeting are dynamic content distribution, server switching, and warm standbys. This is research work that I am doing with Prof. Mendel Rosenblum at Stanford. More info (papers, presentations) is at *http://www.thevmatrix.com*

Taking SysNav Open Source

Christian L Pearce
pearcec@commnav.com
sysnav.commnav.com

SysNav started out as a closed source project for managing servers via a portal infrastructure. It consists of storing configuration information about machines and what components they would like managed. This information is held in LDAP and translated into cfengine files and configuration files by the middle layer. Then the back end takes these configuration files and executes them via cfengine. This framework will install, upgrade, and configure components automatically based on the information stored in LDAP. SysNav is going through a transition. It is CommNav's goal to take the back end and the middle layer and form an open source meta-project. We, at CommNav, feel the community will benefit from the project and other sub-projects that will be generated out of taking SysNav open source. Collaboration has already begun internally and will be released in 2003. Please see *http://sysnav.commnav.com* for more information.

The Configuration Monitoring and Reporting Environment

Xev Gittler
usenix-lisa@schore.org

The Configuration Monitoring and Reporting Environment (CMRE) is designed to collect configuration data from all our systems and then correlate and report on the information. This allows us to understand exactly the state of our systems, from OS levels and hardware, to software installed and patches, to security and audit problems, to standards conformance. We collect this data and save it for historical data collection (via CVS), as well as upload a significant portion to a database to do reporting across the company at various levels of detail. We also combine this with our performance monitoring to identify the most over- and under-utilized systems.

Software for Optimal Time to Patch

Adam Shostack
adam@homeport.org

Following on research presented in the refereed papers track, Adam has founded a company to build decision support software for IT departments to find the optimal time to install patches, maximizing their uptime and reliability. Adam is interested in talking to IT managers who measure uptime and security.

What?? Another &%#!'ing Backup Package?

James O'Kane
jo2y@midnightlinux.com

I'll talk briefly about why I'm writing yet another backup application and why this one will be newer, better, different. So cool, that hopefully you'll forget why you thought digital watches were a pretty neat idea.

Return of the Son of the Bride of Conserver (aka Conserver 8.0.0)

Bryan Stansell

The conserver application was developed by Tom Fine in 1990 to allow multiple users to watch a serial console at the same time. Despite its indispensability, many sysadmins aren't aware of it. Conserver can log console output, allows users to take write access of a console (one at a time), and has a variety of bells and whistles to accentuate that basic functionality. The idea is that conserver will log all your serial traffic so you can go back and review why something crashed, look at changes (if done on the console), or tie the console logs into a monitoring system. With multiuser capabilities you can work on equipment remotely, collaborate with others, and mentor junior admins. (See Fine and Romig, LISA IV Conference Proceedings, 97-100.)

Since then, many enhancements have been added. The current conserver.com version (7.2.4) also includes basic SSL support so that, assuming you have a network connection, you can securely interact with any of the equipment from home or wherever. The next version will have yet another slew of enhancements, including complete SSL support and a new config file format. In this WIP, I'll give you the scoop on the latest features and solicit you for additional cool ideas for the code and a possible future paper.

rcs.mgr

John Rowan Littell
littejo@earlham.edu
www.earlham.edu/~littejo/

rcs.mgr is a basic, self-contained configuration manager that wraps the RCS process for textual configuration files and manages their installation, including setting ownerships and permissions and running any post-installation commands necessary to activate the changes. The script has been in production for 1.5 years. Future developments will include better handling of unauthorized changes and support for per-file editors, allowing the management of non-textual files.

Infinite Scalability Distribution

Doug Hughes
doug@gblx.net

I have a multicast distribution program that has been "under development" for about two years now. It puts a sequence number on each datagram and uses selective retransmission from the receiver to the sender to get missing sequence numbers. It also uses PGP signatures on each whole "package" for authenticity and for integrity; this also allows building of a web of trust. The "file" program is used to determine how to process the received item on each

receiving host. Each "distrib item" is signed with PGP and multicasted to all listening clients on a well-defined port. Responses can be collated in many different ways: syslog, mail, tcp socket, file, etc. The software provides distribution and an extensible framework upon which to build. A distribution server can also be used as a generic request repository. A peer-to-peer network of senders and requestors can thus be built easily.

Improving Productivity

Jeremy Mates
jmates@sial.org

http://www.sial.org/code/perl/modules/

Sial::Apache::ImageShow (1.2)

The talk is available at *http://www.sial.org/talks/productivity/* with pointers to the script.

Peg's Notes: Jeremy showed true WIP spirit by developing this presentation moments before he was to go on the stage! His HUGE contribution to productivity allows users to see all their favorite daily comics on ONE page!

Administering Self-Secure Devices

A. Chris Long
aclong@ece.cmu.edu

Suppose you had a host-based and a network-based IDS on every computer in your enterprise. How would you manage them? The "self-secure devices" we are developing are disk drives and NICs that include security measures, such as monitoring for changes to system files and virus traffic. I am in the early stages of designing the user interface for a system administrator to configure, monitor, and control self-secure devices.

CONNECT WITH USENIX

MEMBERSHIP AND PUBLICATIONS

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

http://sagewire.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.

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

;login:

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

POSTMASTER
Send Address Changes to *;login:*
2560 Ninth Street, Suite 215
Berkeley, CA 94710

PERIODICALS POSTAGE
PAID
AT BERKELEY, CALIFORNIA
AND ADDITIONAL OFFICES