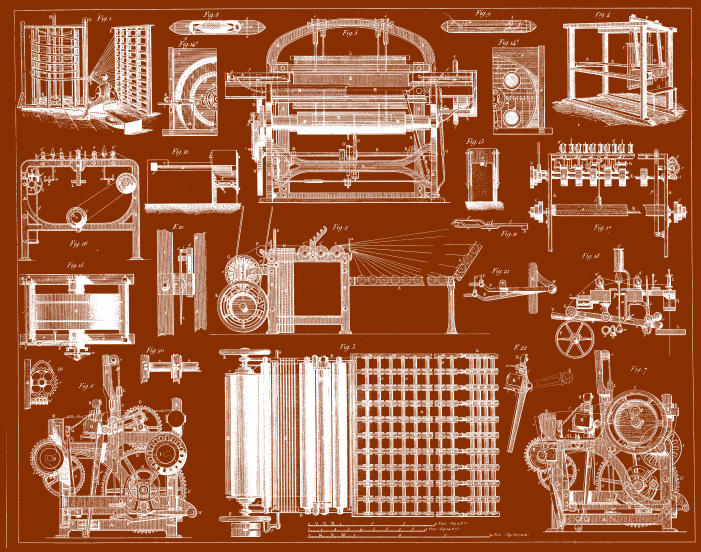
DECEMBER 2010 VOLUME 35 NUMBER 6

;login:

THE USENIX MAGAZINE



USENIX
The Advanced Computing Systems Association

USENIX Upcoming Events

9th USENIX Conference on File and Storage Technologies (FAST '11)

Sponsored by USENIX in cooperation with ACM SIGOPS

FEBRUARY 15–18, 2011, SAN JOSE, CA, USA
http://www.usenix.org/fast11

Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE '11)

Co-located with NSDI '11

MARCH 29, 2011, BOSTON, MA, USA
http://www.usenix.org/hotice11

4th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '11)

Co-located with NSDI '11

MARCH 29, 2011, BOSTON, MA, USA
http://www.usenix.org/leet11

8th USENIX Symposium on Networked Systems Design and Implementation (NSDI '11)

Sponsored by USENIX in cooperation with ACM SIGCOMM and ACM SIGOPS

MARCH 30–APRIL 1, 2011, BOSTON, MA, USA
http://www.usenix.org/nsdi11

European Conference on Computer Systems (EuroSys 2011)

Sponsored by ACM SIGOPS in cooperation with USENIX

APRIL 10–13, 2011, SALZBURG, AUSTRIA
http://eurosys2011.cs.uni-salzburg.at

13th Workshop on Hot Topics in Operating Systems (HotOS XIII)

Sponsored by USENIX in cooperation with the IEEE Technical Committee on Operating Systems (TCOS)

MAY 8–10, 2011, NAPA, CA, USA
http://www.usenix.org/hotos11
Submissions due: January 15, 2011

3rd USENIX Workshop on Hot Topics in Parallelism (HotPar '11)

MAY 26–27, 2011, BERKELEY, CA, USA
http://www.usenix.org/hotpar11
Submissions due: January 16, 2011

2011 USENIX Federated Conferences Week

JUNE 12–17, 2011, PORTLAND, OR, USA

Events include:

3rd Workshop on Hot Topics in Storage and File Systems (HotStorage '11)

JUNE 14, 2011
http://www.usenix.org/hotstorage11
Submissions due: March 9, 2011

3rd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '11)

JUNE 14–15, 2011
http://www.usenix.org/hotcloud11

2011 USENIX Annual Technical Conference (USENIX ATC '11)

JUNE 15–17, 2011
http://www.usenix.org/atc11
Submissions due: January 12, 2011

2nd USENIX Conference on Web Application Development (WebApps '11)

JUNE 15–16, 2011
http://www.usenix.org/webapps11
Submissions due: January 21, 2011

20th USENIX Security Symposium (USENIX Security '11)

AUGUST 10–12, 2011, SAN FRANCISCO, CA, USA
http://www.usenix.org/sec11
Submissions due: February 10, 2011

Workshops co-located with USENIX Security '11 (as of 11/10):

2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE '11)

AUGUST 8–9, 2011
http://www.usenix.org/evtwote11
Submissions due: April 20, 2011

23rd ACM Symposium on Operating Systems Principles (SOSP 2011)

Sponsored by ACM SIGOPS in cooperation with USENIX

OCTOBER 23–26, 2011, CASCAIS, PORTUGAL
http://sosp2011.gsd.inesc-id.pt
Submissions due: March 11, 2011

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

contents

;login:
VOL. 35, #6, DECEMBER 2010

USENIX ASSOCIATION
2560 Ninth Street,
Suite 215, Berkeley,
California 94710
Phone: (510) 528-8649
FAX: (510) 548-5738

http://www.usenix.org
http://www.sage.org

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

;login: (ISSN 1044-6397) is published bi-monthly by the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

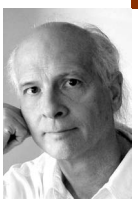
$90 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $125 per year.

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.

RIK FARROW

musings



Rik is the Editor of *;login:*.

rik@usenix.org

I'VE BEEN ACCUSED, RIGHTLY, OF BEING pessimistic about computer security, and recent events have only increased that pessimism. But rather than tire you with my grumblings, I thought I would take a dispassionate look at computer security as it exists today and make positive suggestions about what you might do, whether in your professional or personal lives.

I'll start out with something you might find surprising, considering the source: if you, or people you know or work with, use Windows XP, convince them to upgrade. The same goes for people using anything earlier than Server 2008.

Microsoft began its Trustworthy Computing Initiative in 2002 and has paid much more attention to security in recent years. Some of the fruits include more reactive security measures, such as DEP (data execution prevention) and ASLR (address space layout randomization), although these are not used in all applications. Internet Explorer 7 prior to SP1 is one of those applications that is not protected with either DEP or ASLR for application compatibility, but later versions are, as is IE8.

Both IE7 and IE8 also rely on Integrity Levels [1], an ACL mechanism where less trusted processes, such as Web browsers, get run with a low integrity level. Processes with low integrity levels have limited or no access to files, processes, or other objects (e.g., registry keys and named pipes) at higher integrity levels—which means, most of the system.

These are good things. I kept hearing from my friends in security that Windows had gotten a lot more secure—but they wouldn't or couldn't provide strong evidence that these mechanisms actually help. Then I learned from Niels Provos, whose Google team searches the Web for malicious sites, that it was much more difficult for most exploits to work with IE7 or IE8. While his team's goal is to find pages that lead to exploits on any version of Windows, I found this interesting news, as they actually test hundreds of millions of pages in their Windows equipped sandboxes (see "The Nocebo Effect," p. 18).

Crispin Cowan, the inventor of stack canaries, also known for Immunix and AppArmor, began working for Microsoft in 2008. Cowan spoke at the 2010 USENIX Security Symposium, allegedly

about the security features of Windows 7 but actually about how Microsoft had sometimes been the first vendor to include new security features. I have it on good authority [2] that such talk is security theater, but you can watch the video of his presentation and decide for yourself [3]. You can also read the summaries of his talk and that of Roger Johnston, the person who describes Cowan's talk as security theater, in this issue.

One point Cowan made that really struck me was this: in 2010, the number of applications that needed administrative privileges to run had been reduced from 900,000 to just 180,000 (49 minutes into the video). I was dumbstruck.

I always knew there were lots of Windows applications, but that there are nearly two hundred thousand that need to run as root just astounded me. Cowan works as a senior project manager on User Access Control, what he called "the moral equivalent of sudo." So running these apps requires sudo to the admin group. You might not need to run any of these apps, but now you know what UAC is doing for you, or allowing these apps to do to your system.

Dark Side

So Windows has its dark side. We've always known that. The need for running apps with privileges has to do with the history of Windows NT, which Cowan also covered earlier in his talk. In 1995, NT had no applications, so by adding the Windows 32-bit API libraries to NT, there were suddenly many thousands of applications. Unfortunately, there were also many, many millions of lines of old code, not written with security in mind.

We still have patch Tuesday, as well as security excitement for all operating systems. None of this will be going away, as the number of programmers capable of writing mostly secure programs is extremely limited. At a past security symposium, a speaker suggested during a WiP that there were only two such programmers, Wietse Venema and Daniel Bernstein. I think this is an exaggeration, leaving out other outstanding programmers. But the point is that most programmers are not particularly good, and certainly not good at security.

At the same time, people are encouraged to write programs. Microsoft's very success is tied to its vast number of applications. But so is Microsoft's greatest weakness: maintaining backward compatibility so that it doesn't lose this asset. This is a problem for all systems today, as adding software—say, a cool PHP-powered Web site—to a server is easy. None of this is news.

Is Windows 7 safe to use? It is safer, but not safe. For example, the ZeuS botnet has been in the news as I write [4], and this criminal tool includes exploits for Vista and 7 [5].

Being safe on the Web today is still difficult. I suggest booting Linux (or a BSD) from a CD or write-protected USB stick and using this for your must-be-secure browsing, such as banking. Next best is to avoid the most popular platforms, such as Windows and Mac, and stick with something obscure like FreeBSD (if you can get the financial site to work with Firefox). Note that malware is starting to appear on smartphones, so banking online using your handheld device may not be safe either. Am I paranoid? Yes, I am paranoid, especially when I recently learned that a security friend lost $35,000 from his bank account.

If you are running a server, consider using tools such as SELinux to sandbox the server's applications. While type-enforcement mechanisms will not protect a server from bugs within itself, such as SQL injection, it will prevent exploits from escaping the application in most instances. Then again, Linux kernel exploits may be designed to bypass, or even abuse, SELinux in the exploit [6].

A Better Sandbox

Robert Watson, a key contributor to the FreeBSD kernel and, by association, to Mac OS X, has created a different way of sandboxing applications. Working with a colleague at the University of Cambridge and two people at Google, Watson developed Capsicum, a capability-based sandbox.

I found a couple of things interesting about Capsicum. First, it attempts to make life simpler for the programmer. Its basic principle involves severely limiting access to the operating system's namespace: files, IPC, shared memory, and even network access. The capabilities used in Capsicum are, for example, open files. Once an application enables Capsicum, only already open files, or files within an already open directory, can be accessed. Capsicum can also work in programs that split privilege levels, such as OpenSSH sshd. The privileged part of a Capsicum-enabled program maintains access to the system namespace and can share capabilities, such as open files or network connections, with the constrained fork of the program.

Capsicum also places the security policy for an application within the application itself. Using SELinux or Microsoft's Integrity Levels and ACLs means that a large portion of an application's security policy exists in system configuration—for example, in Type Enforcement and File Context rules in SELinux. With Capsicum, upgrading a program's security policy is done by upgrading the program, without needing to change system security policy.

The Lineup

Robert Watson, Jon Anderson, Ben Laurie, and Kris Kennaway start off this issue by explaining Capsicum in more detail. Their article compares Capsicum with other forms of sandboxing, as used in Chromium, as well as providing an example of securing tcpdump by dropping privileges after they are no longer needed.

Next up, Moheeb Rajab, working with a team of Google security engineers, updates us on a trend in malware. I've asked Niels Provos to write about his team's activities in the past, and this time Rajab explains that their data clearly shows an increase in the amount of exploits designed to trick people into installing fake antiviral software. I know people who have been fooled by this; I convinced one of them to install Linux instead of reinstalling Windows and buying AV.

Dan Geer, who as Invited Talks co-chair at USENIX Security help to serve up an excellent list of speakers, reprises his own invited talk. Geer ponders the problem created by standards, especially when the protocols they describe are so complex that everyone ports the reference implementation. The result is a form of monoculture, where most, or even all, systems include the same bugs. We have seen this most recently in TLS renegotiation, a protocol that appears in embedded systems as a security feature.

I saved the hardware security article for last, as some may find it a deep dive. Matthew Hicks and co-authors write about their design for working around hardware that includes suspicious circuits. In an earlier paper [7], King et al. showed how they could add circuits to a SPARC CPU that provided a foothold in a system that could easily lead to complete compromise. In this article, Hicks et al. explain how to detect potentially malicious circuits and provide workarounds in software, allowing a system found to include malicious circuits to be patched in the field. I do recommend that you read this article if you wish to learn more about the problem of hardware that may include malicious designs, and a possible solution.

David Blank-Edelman takes us on a utilitarian journey of modules that provide, well, utilities. How practical, and useful as ever.

Peter Galvin begins to explore alternatives to Solaris in the first of a two-part column. In this issue, Galvin compares and contrasts what he considers the most likely contenders to Solaris in enterprise-level computing: Linux and AIX. Yes, I wrote AIX, and don't write off this unusual UNIX variant without a closer look.

Dave Josephsen explores Ganglia, a tool for monitoring clusters of systems. Josephsen obviously likes Ganglia (enough), partially for the ease of configuring clients and for its lightweight footprint.

Robert Ferrell ponders the intent of a hardware manufacturer who is selling CPUs with key features disabled—but will enable them if you are willing to pay a ransom.

Elizabeth Zwicky gets into the Christmas spirit, including reviews of two cooking books and a LEGO book, as well as two technical books. I had also read *Cooking for Geeks* and would have called it *Cooking for Hackers*, as it is full of the type of details I wanted to find out years ago. I also wrote a review on a book you may consider buying as a gift for someone, *Your Money: The Missing Manual*. Sam Stover reviewed our only security book this time, *Inside Cyber Warfare*, and it sounds like an interesting and quick read.

We have reports on the 2010 USENIX Security Symposium, as well as reports for three of the seven workshops that were co-located with Security. We also have a summary of NSPW, an interesting security workshop with very limited attendance.

I am not really worried about depressing you when it comes to news about security. If you aren't depressed, something is wrong with you, or you just haven't been paying attention.

Stuxnet, a bit of very competently designed malware aimed specifically at Siemens S7 control systems used in Iran, has been in the news as I muse [8]. Stuxnet, spread via USB sticks, includes *four* Windows zero-day exploits and two signed device drivers, using keys stolen from two companies in Taiwan. The malware is carefully written, so that it never crashes the systems it infects, never communicates with its creators, and only causes havoc when it detects it is running on the S7 systems installed in very specific applications.

In other words, Stuxnet appears to be the first shot in a "cyber war"—a term I hate, but I don't know what else fits. And now that the cat is out of the bag, I expect we will begin to see copycat attacks take down other SCADA-controlled systems, with the developed world, particularly the United States, being particularly vulnerable.

When computer systems were first used, they were terribly expensive and carefully isolated systems. As this changed in the 1980s, people were just

happy to have computers they could afford. In the 1990s, prices of systems began to plummet, with the first under-$1000 system appearing around 2000. Now you can buy a netbook for under $400 and smartphones more powerful than a 1980 Cray. None of this history includes a mandate for secure computing.

Building secure computer systems requires a complete redesign of both software and hardware, and this isn't going to happen overnight. I do see some things I like, such as SeL4, type-safe languages, and experimental multicore designs such as the Single Chip Cloud. But restarting computer science, where security is built in and unavoidable, instead of an added-later feature, is still years away.

REFERENCES

[1] Integrity Levels, used in Vista and Win2008 and later: https://secure.wikimedia.org/wikipedia/en/wiki/Mandatory_Integrity_Control.

[2] Roger Johnston, Editor's Notes, *Journal of Physical Security*, vol. 4, no. 2 (2010): http://jps.anl.gov/v4iss2.shtml.

[3] Crispin Cowan, "Windows 7 Security from a UNIX Perspective": http://www.usenix.org/events/sec10/stream/cowan/index.html.

[4] ZeuS bot installed using LinkedIN spam: http://www.thestar.com/news/gta/crime/article/869704--email-attack-targeting-linkedin-users-termed-largest-ever.

[5] ZeuS for Windows 7: http://www.secureworks.com/research/threats/zeus/?threat=zeus.

[6] Cheddar Bay Linux kernel exploit: http://lwn.net/Articles/341773/.

[7] Samuel T. King et al., "Designing and Implementing Malicious Hardware," Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '08), April 2008.

[8] Stuxnet May Be Targeting Iran's Nuclear Sites: http://www.bloomberg.com/news/2010-09-24/stuxnet-computer-worm-may-be-aimed-at-iran-nuclear-sites-researcher-says.html.

ROBERT N.M. WATSON,
JONATHAN ANDERSON, BEN LAURIE,
AND KRIS KENNAWAY

introducing Capsicum: practical capabilities for UNIX

APPLICATIONS ARE INCREASINGLY turning to *privilege separation*, or *sandboxing*, to protect themselves from malicious data, but these protections are built on the weak foundation of primitives such as chroot and setuid. Capsicum is a scheme that augments the UNIX security model with fine-grained *capabilities* and a sandboxed *capability mode*, allowing applications to dynamically impose capability discipline on themselves. This approach lets application authors express security policies in code, ensuring that application-level concerns such as Web domains map well onto robust OS primitives. In this article we explain how Capsicum functions, compare it to other current sandboxing technologies in Linux, Mac OS, and Windows, and provide examples of integrating Capsicum into existing applications, from tcpdump and gzip to the Chromium Web browser.

Robert N.M. Watson is a PhD candidate at the University of Cambridge Computer Laboratory. His PhD research is in operating system security extensibility. Prior to joining the Computer Laboratory, he was a Senior Principal Scientist at McAfee Research, now SPARTA ISSO, where he directed commercial and government research and development projects in computer security, including the TrustedBSD MAC Framework now used for access control in FreeBSD, Juniper Junos, Mac OS X, and Apple iOS. His research interests include operating system security, network stack performance, and the evolving software-hardware interface. Mr. Watson is also a member of the board of directors for the FreeBSD Foundation, a 501(c)(3) non-profit supporting development of FreeBSD, a widely used open source operating system.

robert.watson@cl.cam.ac.uk

Jonathan Anderson is a PhD student in the University of Cambridge Computer Laboratory. His research interests include operating system security and privacy in distributed social networks.

jonathan.anderson@cl.cam.ac.uk

Ben Laurie works on security, anonymity, privacy, and cryptography and thinks object capabilities are the best hope we've got. He currently splits his time between the Applied Security group at Google and working on the Belay project at Google Research.

benl@google.com

Kris Kennaway is a Senior Site Reliability Engineer at Google. He is a former FreeBSD Security Officer and a former member of the FreeBSD Core Team. His interests include computer security, operating system scalability on multi-core hardware, and the design, care, and feeding of large-scale distributed systems. Kris received a PhD in theoretical physics from the University of Southern California in 2004.

kennaway@google.com

Compartmentalization

Today's security-aware applications are increasingly written as compartmentalized applications, a collection of cooperating OS processes with different authorities. This structure, which we term a "logical application" and illustrate in Figure 1, is employed to mitigate the harm that can be done if inevitable vulnerabilities in application code are exploited.

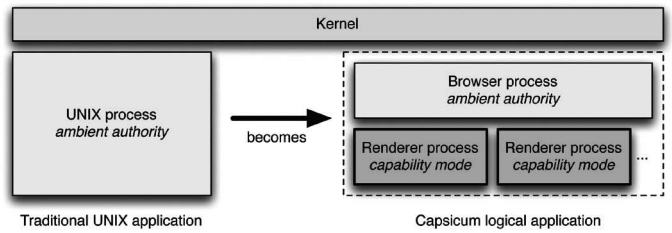


FIGURE 1: CAPSICUM HELPS APPLICATIONS SELF-COMPARTMENTALIZE.

For instance, in the Chromium Web browser, a malicious image that exploits a libpng vulnerability can be confined to a *renderer process* responsible for converting Web content such as HTML and compressed images into pixels. Such a process has less access to OS services such as the file system and network stack than the main *browser process*,

so the damage that can be done by malicious content is limited. Other widespread examples of software using this technique include PackageKit, Apple's Security Server, and OpenSSH's sshd.

Unfortunately, self-compartmentalizing code is very difficult to write, as contemporary commodity operating systems are firmly engrained with the notion of *ambient authority:* applications running with the full authority of the user who launched them. Creating a sandbox thus involves restricting existing access to user- or system-level rights, a process which frequently itself requires system privilege.

Capabilities

At the other end of the authority spectrum are capability systems, such as CMU's Hydra operating system [1], that support true least-privilege discipline in their applications. In such a system, application code can only exercise authority (e.g., access user files) through fine-grained *capabilities*, unforgeable tokens of authority, which have been delegated to it.

Capability systems are designed around delegation, since they allow tasks to selectively share fine-grained rights with other tasks through inheritance and explicit assignment. In this model, the operating system enforces the isolation of tasks and the restriction-associated capabilities, but semantically rich policy—what the capability *means* and who should have access to it—is defined by applications. This separation of mechanism and policy is very useful, and it is one which we sought to enhance on the UNIX platform by the addition of capability features.

Capsicum

Capsicum is a new approach to application compartmentalization. It is a blend of capability and UNIX semantics which, we believe, has some of the best characteristics of both. It allows applications to share fine-grained rights among several rigorously sandboxed processes, but preserves existing UNIX APIs and performance. Capsicum also provides application writers with a gradual adoption path for capability-oriented software design.

DESIGN

Capsicum extends, rather than replaces, standard UNIX APIs by adding new kernel primitives and userspace support code to help applications self-compartmentalize.

The most important new kernel primitives include a sandboxed *capability mode*, which limits process access to all global OS namespaces, and *capabilities*, which are UNIX file descriptors with some extra constraints. The userspace additions include *libcapsicum*, a library which wraps the low-level kernel features and a *capability-aware run-time linker.*

CAPABILITY MODE

Capability mode is a process credential flag set by a new system call, `cap_enter`, available to all users. Once set, the flag is inherited by all descendent processes and cannot be cleared. Processes in capability mode are denied access to global namespaces such as the file system, PIDs and SystemV IPC namespaces.

Access to system calls in capability mode is also restricted: some system calls requiring global namespace access are unavailable, while others are constrained. For instance, sysctl can be used to query process-local information such as address space layout, but also to monitor a system's network connections. We have constrained sysctl by explicitly marking ≈30 of 3000 parameters as permitted in capability mode; all others are denied.

The system calls requiring constraints include sysctl, shm_open, which is permitted to create *anonymous memory objects*, but not named ones, and the openat family of system calls. The *at calls already accept a file descriptor argument as the directory relative to which to perform the open, rename, etc.; in capability mode, they are constrained so that they can only operate on objects "under" this descriptor. For instance, if file descriptor 4 is a capability allowing access to /lib, then openat(4, "libc.so.7") will succeed, whereas openat(4, "../etc/passwd") and openat(4, "/etc/passwd") will not. This allows partial namespace delegation, as shown in Figure 2.

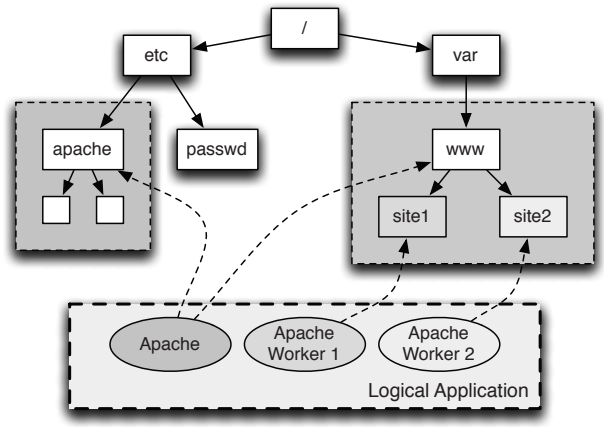


FIGURE 2: AUTHORITY OVER PORTIONS OF THE FILE SYSTEM CAN BE DELEGATED.

CAPABILITIES

In Capsicum, a capability is a type of file descriptor that wraps another file descriptor and constrains the methods that can be performed on it, as shown in Figure 3.

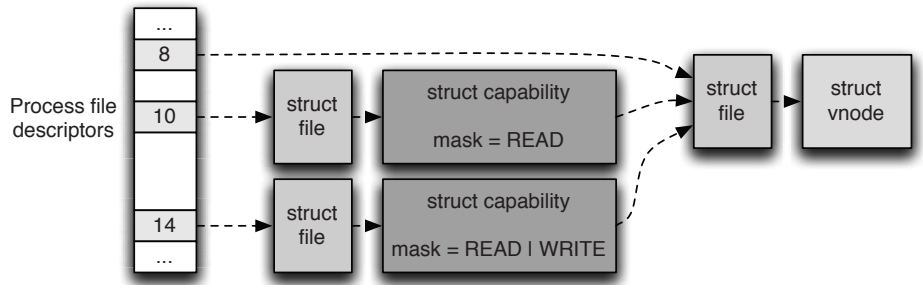


FIGURE 3: CAPABILITIES "WRAP" NORMAL FILE DESCRIPTORS, MASKING THE SET OF PERMITTED METHODS.

File descriptors already have some properties of capabilities: they are unforgeable tokens of authority and can be inherited by a child process or passed between processes that share an IPC channel. Unlike true object capabilities, however, they confer very broad rights as a side effect: even if a file descriptor is read-only, operations on metadata such as fchmod are

permitted. Capsicum restricts these operations by wrapping the descriptor in a capability descriptor, checking the mask of allowable operations whenever the file object is looked up. For instance, when the read system call is invoked with a capability, that capability can only be converted to a file object if its mask includes CAP_READ.

Capabilities are created via the cap_new system call, which accepts an existing file descriptor and a mask of rights as arguments. If the original descriptor is a capability, the result will be a new capability with a subset of the original's rights; applications may always reduce the privilege of a file descriptor, but they may never escalate it. Like other file descriptors, capabilities may be inherited across fork and exec, as well as passed via UNIX domain sockets.

There are approximately 60 rights which a capability can mask, striking a balance between pure message-passing (two rights: send and receive) and MAC systems (hundreds of access control checks). We have selected rights which align with logical methods on file descriptors; some system calls require multiple rights, and calls implementing semantically identical operations require the same rights. For example, pread (read to memory) and preadv (read to a memory vector) both require CAP_READ in a capability's rights mask, while read (read bytes using the file offset) requires CAP_READICAP_SEEK.

Capability rights are checked by fget, the in-kernel function for converting file descriptor numbers into in-kernel references. This strategy—implementing checks at a single point of service deep in the kernel, rather than in several system calls—is repeated throughout Capsicum, providing assurance that no alternate code paths exist which could be used to bypass checks.

Many past security extensions have composed poorly with UNIX security, leading to vulnerabilities. As a result, we disallow privilege elevation via fexecve using setuid and setgid binaries in capability mode. This restriction does not prevent setuid binaries from using sandboxes.

RUN-TIME ENVIRONMENT

Even with Capsicum's kernel primitives, creating sandboxes without leaking undesired resources via file descriptors, memory mappings, or memory contents is difficult. Processes, including libraries they use, may access resources with overly broad rights, or fail to relinquish access when it is no longer needed. Furthermore, introducing robust sandboxing forces fundamental changes to the UNIX run-time environment: even fork and exec rely on global namespaces—process IDs and the filesystem namespace.

libcapsicum therefore provides an API for starting sandboxed processes and ensuring that they only possess authority which has been explicitly delegated to them.

After creating a new process with the descriptor-oriented pdfork, libcapsicum cuts off the sandbox's access to global namespaces via cap_enter. In order to ensure that rights are not accidentally leaked from parent to child, it then closes all inherited file descriptors that have not been positively identified for delegation and flushes the address space via fexecve. Sandbox creation returns a UNIX domain socket that applications can use for inter-process communication (IPC) and for sharing additional rights between host and sandbox.

Starting a process inside a sandbox requires a Capsicum-aware run-time linker, which loads dynamic libraries from read-only directory descriptors rather than the global filesystem namespace. The main function of a program can call lcs_get to determine whether it is in a sandbox, retrieve sandbox state, query creation-time delegated capabilities, and retrieve an IPC handle so that it can process RPCs and receive runtime delegated capabilities. This allows a single binary to execute both inside and outside of a sandbox, diverging its behavior based on its execution environment.

APPLICATIONS

Adapting applications for use with sandboxing is a non-trivial task, regardless of the framework, as it requires analyzing programs to determine their resource dependencies, and adopting a distributed system programming style in which components must use message passing or explicit shared memory rather than relying on a common address space for communication. Capsicum does not solve this problem; what it does do is make it easy for an application writer, having decided where a security boundary should lie, to enforce it by creating a robust sandbox and sharing fine-grained, least-privileged rights with it.

We describe in this article two applications that we have modified to take advantage of Capsicum's features, one small and conceptually simple, tcpdump, and one large and complex, Chromium. For more case study details, please see our 2010 USENIX Security Symposium paper [2].

TCPDUMP

tcpdump provides an excellent example of Capsicum primitives offering immediate security benefits through straightforward changes. Historically, tcpdump has been a breeding ground for serious security vulnerabilities, as it has both root privilege and complex packet-parsing code. It is also a very simple program, however, which lends itself handily to sandboxing: resources are acquired early with ambient system privilege, after which packet processing depends only on open file descriptors.

True privilege dropping for tcpdump is accomplished with eight lines of code, shown in Figure 4. Verifying that unneeded privileges have been dropped can be done with the procstat tool; Figure 5 shows that the rights on STDIN have been appropriately constrained.

```
@@ -1197,6 +1199,14 @@
                        (void)fflush(stderr);
        }
 #endif /* WIN32 */
+       if (lc_limitfd(STDIN_FILENO, CAP_FSTAT) < 0)
+               error("lc_limitfd: unable to limit STDIN_FILENO");
+       if (lc_limitfd(STDOUT_FILENO, CAP_FSTAT | CAP_SEEK | CAP_WRITE) < 0)
+               error("lc_limitfd: unable to limit STDIN_FILENO");
+       if (lc_limitfd(STDERR_FILENO, CAP_FSTAT | CAP_SEEK | CAP_WRITE) < 0)
+               error("lc_limitfd: unable to limit STDERR_FILENO");
+       if (cap_enter() < 0)
+               error("cap_enter: %s", pcap_strerror(errno));
        status pcap_loop(pd, cnt, callback, pcap_userdata);
        if (WFileName =NULL) {
```

FIGURE 4: TCPDUMP DROPS ALL UNNEEDED PRIVILEGE WITH EIGHT LINES OF CODE.

```
PIDCOMM          FD  T  FLAGS      CAPABILITIES  PRO  NAME
1268 tcpdump     0   v  rw------c  fs            -    /dev/pts/0
1268 tcpdump     1   v  -w------c  wr,se,fs      -    /dev/null
1268 tcpdump     2   v  -w------c  wr,se,fs      -    /dev/null
1268 tcpdump     3   v  rw-------               - -  /dev/bpf
```

FIGURE 5: PROCSTAT -FC DISPLAYS CAPABILITIES HELD BY TCPDUMP. IN THE CASE OF STDIN, ONLY FSTAT (FS) IS PERMITTED.

CHROMIUM

Google's Chromium Web browser already uses a compartmentalized multi-process architecture similar to a Capsicum logical application on several operating systems [3], so it is an excellent platform for comparing Capsicum with other sandboxing techniques.

Once the FreeBSD port of Chromium was modified to use POSIX rather than System V shared memory (the former, from the Mac OS X port, is descriptor-oriented and thus permitted in Capsicum sandboxes), approximately 100 additional lines of code were required to limit access to file descriptors inherited by and passed to sandbox processes and to call cap_enter.

The result was a robust sandbox that, unlike porous approaches which require hundreds of lines of handcrafted, security-critical assembly code, could be completed in just two days.

Comparison

A plethora of existing security technologies have been used to construct sandboxes in security-aware applications such as Chromium. Each technology has its place—we do not claim that UNIX users and system integrity policies are obsolete—but each also has significant limitations when used for application sandboxing.

We compare Capsicum with five sandboxing mechanisms already employed by Chromium (see Table 1). Each mechanism is used to split the browser into a main browser process, which draws the browser's chrome and interacts with objects such as files, and several renderer processes, which execute untrusted code to uncompress images, interpret JavaScript, etc.

Operating system	*Model*	*Line count*	*Description*
Windows	ACLs	22,350	Windows ACLs and SIDs
Linux	chroot	605	SUID-root sandbox helper
Mac OS X	Seatbelt	560	Path-based MAC sandbox
Linux	SELinux	200	Type Enforcement sandbox domain
Linux	seccomp	11,301	seccomp and userspace syscall wrapper
FreeBSD	Capsicum	100	Capsicum sandboxing using cap_enter

TABLE 1: SANDBOXING MECHANISMS EMPLOYED BY CHROMIUM

Of the six mechanisms employed by Chromium, two are rooted in Discretionary Access Control (users and permissions), two in Mandatory Access Control (labels and system policies), and two in capabilities (unforgeable tokens of authority which are passed between or inherited by processes).

DISCRETIONARY ACCESS CONTROL

In Discretionary Access Control (DAC), the owners of objects specify what rights other users have on those objects; one common example of DAC is the UNIX permissions scheme. Such protections can be used to constrain application behavior if code runs with the authority of a user—such as "nobody" in traditional UNIX systems—with less privilege than the user running the application.

Chromium uses DAC to construct sandboxes on both Windows and Linux. In both cases, inter-user mechanisms fail to provide effective intra-user protections: the robustness of the sandbox is limited, because every user possesses some ambient authority.

Windows ACLs

On Windows, Chromium uses access control lists (ACLs) and security identifiers (SIDs) to effectively run renderer processes as an anonymous user who cannot access objects which belong to "real" users [3]. The unsuitability of the approach is demonstrated well; the model is both incomplete and unwieldy.

The approach is incomplete because objects which are not associated with any user do not receive the protections afforded to objects with ACLs. Some workarounds are possible—for instance, an alternate, invisible desktop is used to protect the user's GUI environment—but many objects remain completely unprotected, including FAT file systems on USB sticks and TCP/IP sockets. Thus, a "sandboxed" renderer process can communicate with any server on the Internet, or even the user's Intranet via a configured VPN!

The approach is also unwieldy in that many legitimate system calls by the sandbox are denied, and must be forwarded to a trusted process which services them on the sandbox's behalf. This forwarding, filtering, and servicing code comprises most of the 22,500 lines of code in the Windows sandbox module, and all of it is absolutely security-critical.

chroot

Chromium's suid sandbox on Linux also attempts to create a privilege-free sandbox using legacy DAC-based access control; the result is similarly porous, and it brings an additional requirement of system privilege.

In this model, access to the file system is limited to a virtual root directory via chroot, but access to other namespaces, including the network and System V shared memory (where the user's X window server can be contacted), is unconstrained.

This sandboxing mechanism also carries an additional requirement: system privilege is required to initiate chroot, so Chromium includes a SUID-root binary which is responsible for starting sandboxes. Thus, sandboxing can only be done with the permission of the system administrator, and any compromise of the setuid binary would have more disastrous consequences than the browser compromise it attempts to protect against.

MANDATORY ACCESS CONTROL

Mandatory Access Control (MAC) is used to enforce system policies such as "files labeled Top Secret shall only be read by users cleared to at least Top Secret," and "files labeled High Integrity shall only be modified by software labeled at least High Integrity."

MAC systems require policy to be described separately from application code. In the context of Multi-Level Secure systems and intelligence applications, this requirement allows rigorous and auditable control of information flow. In the context of sandboxing for consumer applications, however, it leads to the *dual-coding problem*: policy and code will get out of sync, especially if code is written by a vendor and policy by a distribution, so application writers must choose between false positives (legitimate actions are forbidden) and false negatives (illegitimate actions are permitted). In practice, very broad rights are often conferred to avoid blocking legitimate actions.

Furthermore, applying a MAC policy requires the involvement of the system administrator; in order to reduce application authority, system privilege is required. Users are, thus, only protected by MAC if the system administrator has already installed a policy for the software they run, and applications cannot dynamically reconfigure their sandboxes.

SELinux

Chromium supports MAC-based compartmentalization on Linux via an SELinux Type Enforcement policy [4]. We acquired such a policy, not from the Chromium repository, but from the Fedora project, a Linux distribution. Since code and policy come not just from different authors but from different organizations, the dual-coding problem may be expected to be severe.

Compounding the general dual-coding problem further, SELinux policies are so flexible and fine-grained that they are typically written using coarse-grained macros. As an example of one or both of these problems, the Fedora reference policy for Chromium assigns very broad rights, such as the ability to access the terminal device and read all files in /etc.

The requirement for system privilege in defining new policy and types means that Chromium cannot adapt its sandboxes to create new ephemeral security domains for each new website that is visited. For instance, Fedora's policy creates a single SELinux dynamic domain, `chrome_sandbox_t`, which is shared by all sandboxes. Thus, malicious code from evil.com is not prevented from interfering with the renderer process for bank.com.

Mac OS X Sandbox

Chromium also uses a MAC-based framework on Mac OS X to create sandboxes. The Mac OS X sandbox system allows processes to be constrained according to a Scheme-based policy language [5]. It uses the BSD MAC Framework [6] to check application activities against the compiled policy, which can express fine-grained constraints on the file system but, again, coarse all-or-nothing constraints on other namespaces, such as POSIX shared memory.

The Seatbelt-based sandbox model is less verbose than other approaches, but like all MAC systems, security policy must be expressed separately from code, which can lead to inconsistencies and vulnerabilities. Chromium's policy, while restricting access to the global filesystem namespace, allowed access to filesystem elements such as font directories.

CAPABILITIES

The third category of compartmentalization techniques contains *capability*-based approaches. As was mentioned above, capabilities are unforgeable tokens of authority which can be passed between processes, supporting a delegation-oriented security policy.

The UNIX file descriptor is an example of a capability-like object: an application cannot create one without the help of the OS kernel, and once created, it can be shared with other processes, which can then perform system calls such as read and write on it, even if those processes do not have permission to open the file for themselves. UNIX file descriptors are not well-formed capabilities, however. One serious problem with file descriptors is that they are very coarse: a descriptor may allow a process to fchmod the file it points to, even if it was opened with O_RDONLY. Thus, both of the following approaches further limit the authority that a file descriptor conveys and cut off ambient authority.

seccomp

One capability-oriented approach to sandboxing is Linux's seccomp. This is an optionally available mode which denies access to all system calls except read, write, and exit. Processes sandboxed in this way are quite rigorously confined, but only the very simplest applications can use the mode directly; in order to interact meaningfully with the user, network, file system, etc., significant scaffolding code is required to forward system calls, as in the case of the Windows sandbox. Like its Windows counterpart, the Chromium seccomp sandbox contains over a thousand lines of handcrafted, security-critical assembly code to set up sandboxing, implement system call forwarding, and craft a basic security policy (which, incidentally, is default-allow for all filesystem reads; a more complex policy would be even more unwieldy).

Capsicum

Capsicum brings capability concepts to UNIX, allowing sandboxes to be rigorously confined while still able to use capability-oriented UNIX APIs with full UNIX performance.

The modifications required to implement Chromium sandboxing on Capsicum are almost trivial—approximately 100 lines of code—yet they are more robust and flexible than other approaches which require hundreds or even tens of thousands of lines. Furthermore, in contrast to approaches that require system call interception and forwarding, sandboxed processes can operate on file descriptors, and the objects like shared memory which they refer to, with almost no performance degradation.

PERFORMANCE

Typical operating system security benchmarking is targeted at illustrating zero or near-zero overhead in the hopes of selling general applicability of the resulting technology. Our goal is slightly different: application writers have already accepted significant overheads in order to adopt compartmentalization, so we seek to significantly improve security while keeping comparable performance.

Capsicum's capability mode and capabilities are designed to offer native UNIX performance for common operations, as frequently performed operations such as read and write are performed directly on capabilities. Likewise, directory descriptor delegation allows whole UNIX subtrees to be delegated to sandboxes, avoiding message passing on file open in many common cases. This approach is, fundamentally, a hybrid approach, combining elements of the UNIX OS model with a capability system: UNIX would offer unfettered access to the entire file system with privilege, and a capability system might rely on message-passing interposition to filter namespaces, imposing message-passing overhead on common operations.

Detailed performance results, as well as discussion of trade-offs between security and performance, can be found in our USENIX Security paper [2], but suffice it to say that Capsicum primitives are generally as fast as, and sometimes faster than, current UNIX primitives. Performance remains a critical area of research, however; while Capsicum may be cleaner and more efficient for existing privilege-separated applications, adapting further applications will perpetuate current security vs. performance trade-offs. Finding new approaches to improving security performance in the UNIX model is a key concern going forward.

Conclusion

Capsicum is a blending of capability-oriented security with UNIX APIs and performance. Capsicum provides OS foundations that applications can use to compartmentalize themselves with stronger confinement properties and, in some cases, better performance than existing sandboxing techniques. Capsicum is not a replacement for Discretionary or Mandatory Access Control, but we believe that it is superior to them as a platform for application self-compartmentalization.

Much still remains to be done—in some ways, Capsicum is just a platform for more interesting research in systems, programming, and UI security—but we believe that this is a very promising first step.

The Capsicum API and FreeBSD-based prototype are both available today under a BSD license, and more information can be found at http://www.cl.cam.ac.uk/research/security/capsicum/. Capsicum is intended for inclusion in mainline FreeBSD 9.

ACKNOWLEDGMENTS

The authors wish to gratefully acknowledge our sponsors, including Google, Inc., the Rothermere Foundation, and the Natural Sciences and Engineering Research Council of Canada. We would further like to thank Mark Seaborn, Andrew Moore, Joseph Bonneau, Saar Drimer, Bjoern Zeeb, Andrew Lewis, Heradon Douglas, Steve Bellovin, and our anonymous reviewers for helpful feedback on our APIs, prototype, and paper, and Sprewell for his contributions to the Chromium FreeBSD port.

REFERENCES

[1] E. Cohen and D. Jefferson, "Protection in the Hydra Operating System," *SOSP '75: Proceedings of the Fifth ACM Symposium on Operating Systems Principles* (ACM, 1975), pp. 141–60.

[2] R.M. Watson, J. Anderson, B. Laurie, and K. Kennaway, "Capsicum: Practical Capabilities for UNIX," in *Proceedings of the 19th USENIX Security Sympoisum* (USENIX, 2010), pp. 29–45.

[3] C. Reis and S.D. Gribble, "Isolating Web Programs in Modern Browser Architectures," *EuroSys '09: Proceedings of the 4th ACM European Conference on Computer Systems* (ACM, 2009), pp. 219–32.

[4] P. Loscocco and S. Smalley, "Integrating Flexible Support for Security Policies into the Linux Operating System," *Proceedings of the FREENIX Track: USENIX Technical Conference* (USENIX, 2001), pp. 29–42.

[5] "The Chromium Project: Design Documents: OS X Sandboxing Design": http://dev.chromium.org/developers/design-documents/sandbox/osx-sandboxing-design.

[6] R. Watson, B. Feldman, A. Migus, and C. Vance, "Design and Implementation of the TrustedBSD MAC Framework," in *Proceedings of the 3rd DARPA Information Survivability Conference and Exhibition (DISCEX)* (IEEE, April 2003).

MOHEEB ABU RAJAB, LUCAS BALLARD,
PANAYIOTIS MAVROMMATIS,
NIELS PROVOS, AND XIN ZHAO

the nocebo* effect on the Web: an analysis of fake anti-virus distribution

**From the Latin, meaning "I will harm."*

IN RECENT YEARS, PEOPLE HAVE BECOME more aware of malware threats to their computer systems. The common advice to computer users is to install virus and malware detection. This advice has even been codified in Microsoft's Security Center, which provides prominent warnings when such protection is missing.

On the other hand, personal computer systems are lucrative targets for adversaries that compromise computers to steal and monetize sensitive information. As computer systems have become more difficult to compromise, social engineering is becoming an increasingly popular attack vector for enticing users to provide the same information without requiring any vulnerability.

Recently, a threat that we call Fake Anti-Virus has emerged. Fake AV attacks attempt to convince users that their computer systems are infected and offer a free download to scan for malware. Fake AVs pretend to scan computers and claim to find infected files—files which may not even exist or be compatible with the computer's OS. Users are forced to register the Fake AV program for a fee in order to make the fake warnings disappear. Surprisingly, many users fall victim to these attacks and pay to register the Fake AV. To add insult to injury, Fake AVs often are bundled with other malware, which remains on a victim's computer regardless of whether a payment is made.

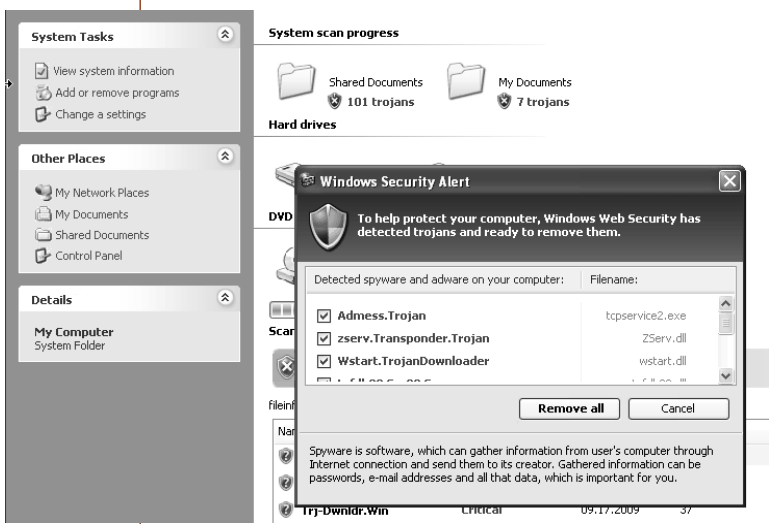


FIGURE 1: A SCREENSHOT OF A FAKE AV SITE. THE BROWSER WINDOW RESEMBLES THE LOOK AND FEEL OF WINDOWS XP.

Moheeb Abu Rajab is a Senior Engineer in the Infrastructure Security group at Google. His research interests are in computer and network security. He received his PhD in computer science from the Johns Hopkins University in 2008.

moheeb@google.com

Lucas Ballard is a member of the Security Team at Google. He received his PhD from The Johns Hopkins University.

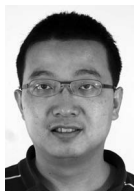
lucasballard@google.com

Panayiotis Mavrommatis is a senior software engineer on Google's security team, fighting Web-based malware. His interests involve computer security and large-scale distributed systems. Panayiotis holds a Master of Engineering from Massachusetts Institute of Technology

panayiotis@google.com

Niels Provos is a Principal Engineer in the Infrastructure Security group at Google. His areas of interest include computer and network security, as well as large-scale distributed systems. He received a PhD from the University of Michigan in 2003, where he studied experimental and theoretical aspects of computer and network security at the Center of Information Technology Integration. He is the author of several popular open source libraries and security tools as well as the book *Virtual Honeypots: From Botnet Tracking to Intrustion Detection*. Dr. Provos currently serves on the USENIX Board of Directors.

niels@google.com



Xin Zhao, PhD, is a software engineer at Google. He holds a doctorate degree in computer science from the University of Michigan. His research interests include security, virtual machines, operating systems, and mobile systems.

xinzhao@google.com

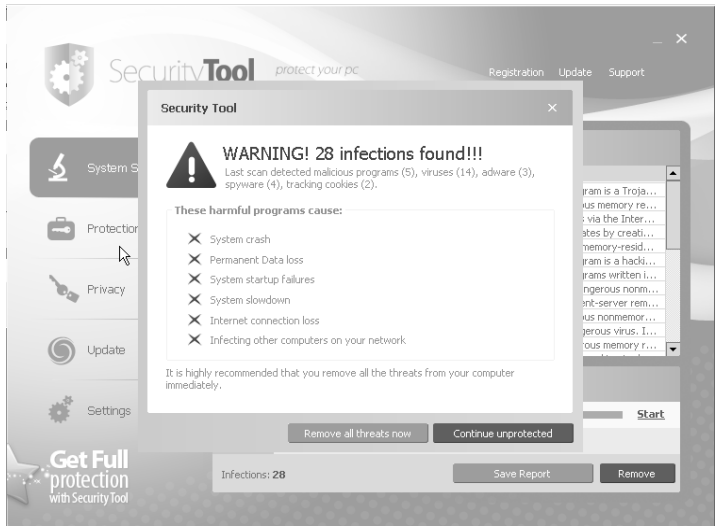


FIGURE 2: A DOWNLOADED FAKE AV BINARY WARNS OF INFECTION AND URGES THE USER TO BUY A PRODUCT.

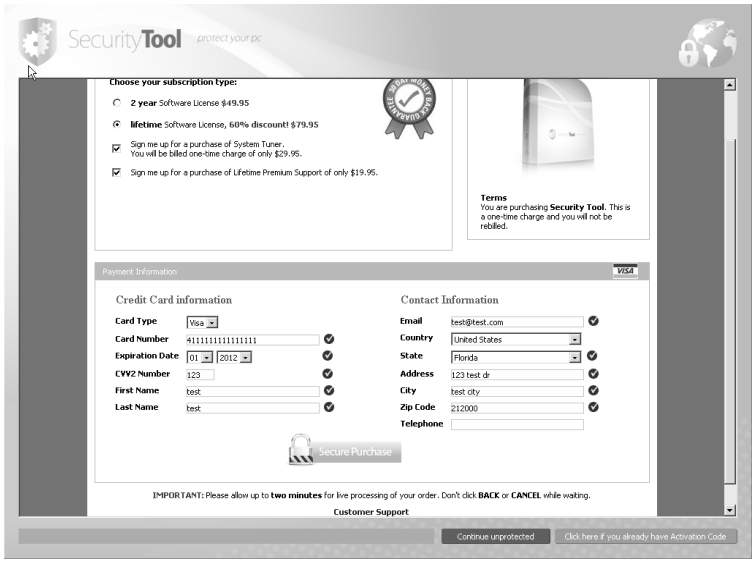


FIGURE 3: A FAKE AV PAYMENT SITE. MANY FAKE AV SITES SHARE THE SAME PAYMENT SITES.

Background

Social engineering attacks scaring users about false insecurities are not new. As early as 2003, malware authors prompted users to download Fake AV software by sending messages via a vulnerability in the Microsoft Messenger Service [5]. We observed the first form of Fake AV attack involving Web sites, e.g., malwarealarm.com, on March 3, 2007. At that time, Fake AV attacks employed simple JavaScript to display an alert that asked users to download a Fake AV executable.

More recent Fake AV sites have evolved to use complex JavaScript to mimic the look and feel of the Windows user interface. In some cases, the Fake AV detects even the operating system version running on the target machine and adjusts its interface to match. Figures 1, 2, and 3 show screenshots representative of Fake AV attacks that we frequently encounter. In Figure 1,

a Web page loads images and text that mimic the appearance of Windows XP. An animated "System scan progress" simulates an ongoing scan for viruses. This is followed by a Windows Security Alert dialog warning the user that various types of malware have been detected. At this point, the Fake AV conveniently provides the user with a button to remove the malware as shown in Figure 2. Clicking the button causes the download and installation of a Fake AV application. This application warns users that their computer is at risk, urging them to buy the full version of the software to "remove all threats." A user who chooses to purchase the software is directed to a payment site (see Figure 3) which asks for credit card information and processes the payment for registering the Fake AV software.

Discovering Fake AV Distributors

We use Google's malware detection infrastructure [2] to discover Web sites that distribute Fake AV software. Briefly, that system uses machine learning to identify potentially malicious Web pages from Google's Web repository. Each page that is flagged by the screening process is further examined by navigating to it with an unpatched Windows virtual machine running an unpatched version of Internet Explorer. Detection algorithms use signals derived from state changes on the virtual machine, network activity, and scanning results of a group of licensed antivirus engines to decide definitively whether a page is malicious.

One of the algorithms is designed to complement our licensed AV engines to specifically detect social engineering attacks, including Fake AV attacks. We do not disclose the details of the detection algorithm, due to the highly adversarial nature of this field. This algorithm is currently used to protect hundreds of millions of Web users from Fake AV attacks and disclosing it may jeopardize this effort.

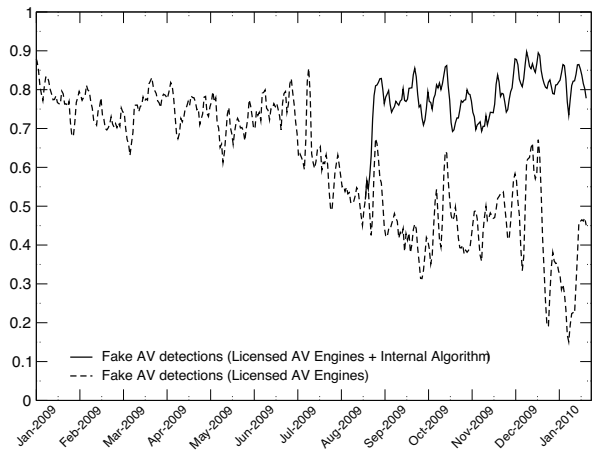


FIGURE 4: FAKE AV DETECTION RATE OVER TIME. INTERNAL ALGORITHMS COUNTER THE INCREASING ABILITY OF ATTACKERS TO EVADE AV ENGINES.

DATA COLLECTION

The data for this article was generated by reprocessing a subset of Web pages that Google's malware detection infrastructure had analyzed between January 1, 2009, and January 31, 2010. Due to the large volume of data, we only reprocessed pages that either resulted in a drive-by download, were convincingly marked as Fake AV, or were otherwise deemed "suspicious" (less than 100% confidence in a page's maliciousness) when they were first visited. Additionally, we scanned a 20% random sample of pages that were originally classified as safe. In total, we reprocessed 240 million pages to establish our data set.

We reprocessed each page using our detection algorithms and virus signatures from mid-February 2010. As Figure 4 shows, our detection rate has improved significantly, reaching up to 90% after we started using our detection algorithm in August 2009.

TERMINOLOGY

Throughout this article we use "Infection domain" to denote a domain that hosts malicious content, including exploits that cause drive-by downloads or content classified as Fake AV. Infection domains are divided into: (1) Exploit domains, which host malicious content that is not a Fake AV, and (2) Fake AV domains, which serve content that was classified as Fake AV using the aforementioned techniques.

Fake AV Distribution Is on the Rise

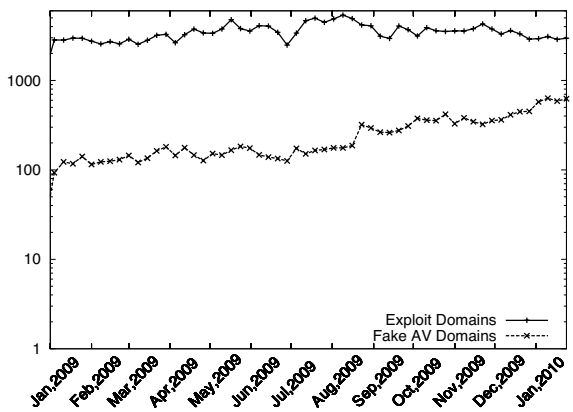


FIGURE 5: TOTAL NUMBER OF NEW INFECTION DOMAINS PER WEEK IN LOG SCALE. FAKE AV DOMAINS EXHIBIT A STEADY UPWARD TREND, WHILE EXPLOIT DOMAINS REMAIN RELATIVELY STABLE OVER TIME.

Figure 5 shows the number of unique first occurrences of both Fake AV and Exploit domains over the course of our study, aggregated by week. Clearly, there is a definitive upward trend in the number of new Fake AV domains. Exploit domains, however, remained relatively stable over time. Indeed, Fake AV accounts for an increasing share of the malware that Google discovers, rising from 3% to 15% over the course of our 13-month study.

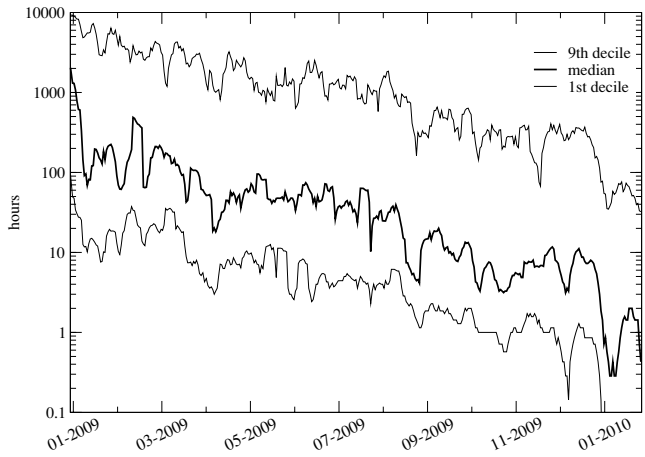


FIGURE 6: LIFETIME OF FAKE AV DOMAINS. THE MEDIAN DROPPED BELOW 10 HOURS IN SEPTEMBER 2009 AND BELOW ONE HOUR IN JANUARY 2010.

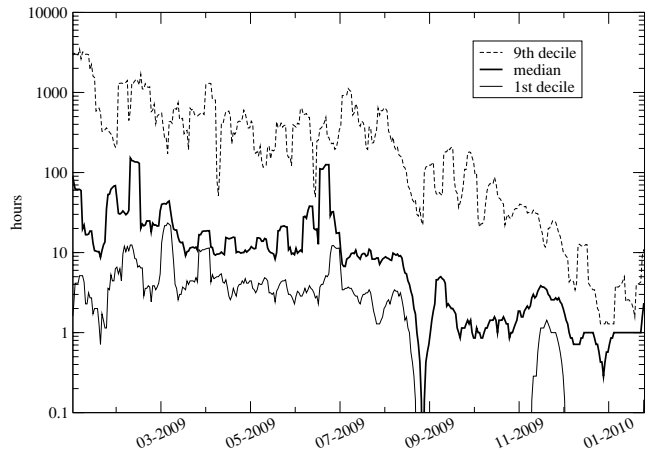


FIGURE 7: TIME TO DETECT FAKE AV DOMAINS.

Analyzing the network characteristics of Fake AV domains revealed strong affinity among groups of Fake AV domains. The 11,480 Fake AV domains mapped to 2,080 IP addresses, with 42% of these IP addresses hosting more than one Fake AV domain.

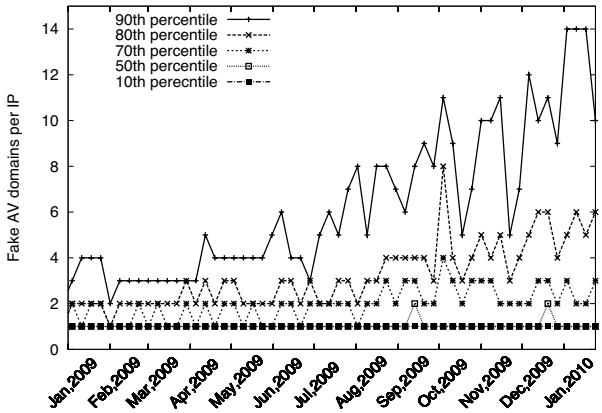


FIGURE 8: PERCENTILES OF THE NUMBER OF FAKE AV DOMAINS (OBSERVED WEEKLY) PER IP ADDRESS

Figure 8 shows an interesting trend: over time, the number of domains served from a single IP address has increased. However, as Figure 6 shows, the lifetime of these domains has actually decreased over time. These trends point to domain rotation, a technique that allows attackers to drive traffic to a fixed number of IP addresses through multiple domains. This is typically accomplished by setting up a number of domains, either as dedicated sites or by infecting legitimate sites, that redirect browsers to an intermediary site under the attacker's control. The intermediary is set up to redirect traffic to a set of active domains, which point to the Fake AV distribution servers.

Domain rotation is likely a response to domain-based detection techniques such as our Safe Browsing API [1]. In fact, we noticed a distinct correlation between our improved ability to detect Fake AVs and the observed lifetime of each domain. Figure 7 shows the trend of our detection time for these domains, measured by the interval between the time at which we would have detected the domain in our baseline data to the actual time our system added the domain to Google's Safe Browsing list. Clearly, the detection time exhibits a downward trend, reflecting an improvement in our ability to detect Fake AV domains quickly after their appearance in our data. This trend is also in line with the reduction in Fake AV lifetime, as depicted in Figure 6.

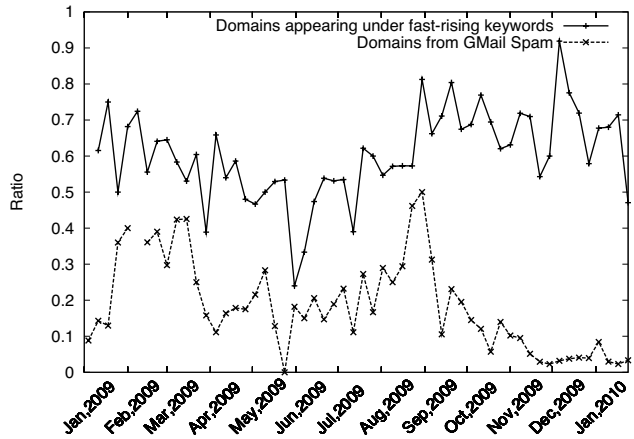


FIGURE 9: RATIO OF FAKE AV DOMAINS TO INFECTION DOMAINS AGGREGATED BY SOURCE OF THE URL. MOST INFECTION DOMAINS ENCOUNTERED ON DOMAINS THAT CONTAIN TRENDING KEYWORDS TEND TO BE FAKE AV DOMAINS.

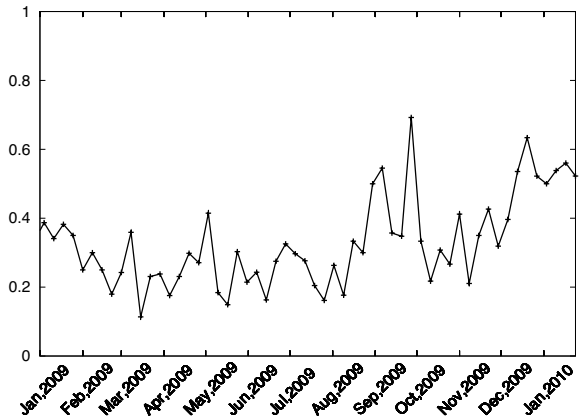


FIGURE 10: RATIO OF FAKE AV DOMAINS TO INFECTION DOMAINS ENCOUNTERED VIA AD NETWORKS. FAKE AV DOMAINS ARE EXHIBITING A RISING TREND TOWARDS AD DISTRIBUTION.

Funneling User Traffic

Fake AV distributors funnel user traffic via a set of Web sites that redirect users to the Fake AV distribution domain. We identified a number of techniques used by Fake AV distributors to lure the users into connecting to the Fake AV site: most notably, setting up dedicated spammy sites that target search engine results optimization for trendy keywords (i.e., Web-search keywords that are fast rising in popularity) and links sent directly via spam emails.

Figure 9 shows the proportion of Fake AV domains to all Infection domains when attributed to these sources. Of note, when our infrastructure identifies Infection domains on recently popular domains, 61% of the time the domain is a Fake AV domain. A smaller percentage of Fake AV domains is observed for domains first seen from Gmail spam. These results indicate that distributors of Fake AV are more successful at targeting domains associated with trending keywords than the distributors of other types of malware.

Another common infection vector for Web-based Malware is ad networks [3]. Our system encounters ad networks in two situations. First, we process URLs from Google Ads' screening pipeline to find and block malicious ads to prevent them being served to users. Second, we encounter ads from non-Google networks while processing other Web pages from Google's index. We examined our data to find Infection domains that use one or more ad networks as intermediaries. Figure 10 shows how often Fake AV domains were delivered via ad networks relative to Exploit domains. Unsurprisingly, as the popularity of Fake AV has increased, so has the number of times Fake AV domains are delivered by ad networks. What is more striking is that, even though Exploit domains are more prominent, we see approximately the same number of Fake AV domains delivered via ads as Exploit domains.

Conclusion

As users are becoming increasingly aware of the need to secure their computers, attackers have been leveraging this awareness by employing social engineering techniques to distribute Fake AV software. Our analysis of Fake AV distribution shows that Fake AV malware now accounts for 15% of all types of malware that we identify. Additionally, we find that Fake AV malware possesses interesting characteristics that distinguish it from typical

Web-based malware. For example, Fake AV domains have more Landing domains funneling user traffic than do other Infection domains.

Fake AV distributors also rely heavily on online advertisements and domains with pages that contain trending keywords. We believe that Fake AV domains have also evolved to use more agile distribution networks that continuously rotate among short-lived domains in an attempt to avoid detection. Despite continuously improving detection and mitigation techniques, Fake AV attacks persist, demanding increased awareness and broader response from the research community at large. For more information, see our publication from USENIX LEET '10 [4].

REFERENCES

[1] Google Safe Browsing API, June 2007: http://code.google.com/apis/safebrowsing/.

[2] N. Provos, P. Mavrommatis, M.A. Rajab, and F. Monrose, "All Your iFRAMES Point to Us," in *Proceedings of the 2008 USENIX Security Symposium,* pp. 1–16.

[3] N. Provos, M.A. Rajab, and P. Mavrommatis, "Cybercrime 2.0: When the Cloud Turns Dark," *Queue*, vol. 7, no. 2 (2009), pp. 46–47.

[4] M.A. Rajab, L. Ballard, P. Mavrommatis, N. Provos, and X. Zhao, "The Nocebo Effect on the Web: An Analysis of Fake Anti-Virus Distribution," in *Proceedings of the 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET' 10),* April 2010: http://www.usenix.org/events/leet10/tech/full_papers/Rajab.pdf.

[5] J. Stewart, "Windows Messenger Popup Spam on UDP Port 1026," June 2003: http://www.secureworks.com/research/threats/popup-spam/, last visited February 18, 2010.

DAN GEER

vulnerable compliance

Milestones: The X Window System and Kerberos (1988), the first information security consulting firm on Wall Street (1992), convenor of the first academic conference on electronic commerce (1995), the "Risk Management Is Where the Money Is" speech that changed the focus of security (1998), the Presidency of USENIX Association (2000), the first call for the eclipse of authentication by accountability (2002), principal author of and spokesman for "CyberInsecurity: The Cost of Monopoly" (2003), co-founder of SecurityMetrics.Org (2004), convenor of MetriCon (2006), author of "Economics & Strategies of Data Security" (2008) and of "Cybersecurity and National Policy" (2010). Advisor variously to global firms FTC, DoJ, DoT, NAS, NSF, USSS, DHS, and five times before Congress. Six startups.

dan@geer.org

A SECURITY PROBLEM MAY BE THEORETI- cal, but when the theoretical becomes practical it is too late for prevention. This essay is not about "responsible disclosure"; its starting point is when disclosure passes the point of inevitability—the instant when the damage control phase begins, even if silently.

Working exploits are cybercrime trade goods, instruments of national policy, or both. But we are here to look at one aspect of this and one only: what to do if a vulnerability is implementation-independent. Vulnerabilities are overwhelmingly dominated by failures of implementation, but that is not our interest.

The designers of what we call the Internet wanted one thing: survivable interoperability. As a network of networks, an Internet neither requires nor expects the construction of some single mechanism under some single control, and that more than one path exists from A to B allows the Internet as we know it blithely to accept random faults, and to route around them. The sum of these two—synthesis by amalgamation plus active fault tolerance—yields survivability, with the side effect that attribution is impossible.

The interoperability goal is inherently harder as interoperability requires out-of-band pre-negotiation of what we commonly refer to as (network) protocol. That is why we have the Internet Engineering Task Force, to standardize protocols in the Internet. Reading directly from "The Tao of the IETF" [1],

> In many ways, the IETF runs on the beliefs of its participants. One of the "founding beliefs" is embodied in an early quote about the IETF from David Clark: "We reject kings, presidents and voting. We believe in rough consensus and running code." Another early quote that has become a commonly-held belief in the IETF comes from Jon Postel: "Be conservative in what you send and liberal in what you accept."

Standing on the foundation of survivability, protocol agreement is the alpha and omega of Internet governance: the "end-to-end" principle [2], which is why the Internet embeds American values and why it works. Governments want instead to embed (their) policy into the transmission fabric, to explicitly eschew an IETF-like process. If you don't want an Internet run by thugs and nannies, now is the time to make yourself heard [3].

So the Internet is composed of an uncoordinated, amalgamational synthesis with active fault tolerance, plus the minimalist coordination of standardized protocols. Standardization is tricky—too early and it kills progress, too late and it just mummifies yesterday's fish. Most longer-lived standards are recognized in place rather than designed from scratch, but the value equation is simply whether standardization at the given moment is more enabling or more disabling. And, of course, any standard has to be implementable ("working code").

I am a strong proponent of diversity of implementations, since implementation flaws are easy to make. Look at CVE [4]. Look at the upward commercial trajectory of the code analysis companies whose entire selling proposition is that implementation errors are easy to make. But the core reason for my fondness for implementation diversity is that implementation diversity quenches cascade failure, but only for implementation-dependent flaws.

There are times when a protocol proves pretty useful and becomes so ubiquitous that we had all better hope that it has no withering flaws since, with scale-up and re-application of the protocol to jobs not foreseen during the standardization phase, flaws will out. If you accept my definition of security, namely "the absence of unmitigatable surprise," then a flaw in a protocol had better be mitigatable, because a flaw in a protocol is guaranteed to be a surprise. In other words, the question on the table is what to do when vulnerability is a consequence of standards compliance, per se.

This is no idle worry. We have a history, and if that history is any guide, then we may as well expect a future little different from the past. A few examples of this phenomenon:

- Between announcement of Kerberos availability in 1988 [5] and the formal retirement of the version 4 protocol 16 years later [6], we have an example of standards compliance implying vulnerability, embodied in an open-source code base as well as an IETF RFC.
- In 2002, Oulu University in Finland found pervasive flaws in version 1 of SNMP, the Simple Network Management Protocol [7]. In this example, it was complexity that deterred the vendor and user communities from avoiding trouble in the first place.
- ASN.1 is more complex than SNMP, complex enough that building a reference implementation is daunting. Microsoft presumably wrote their own and they surely tried hard, but ASN.1 complexity was the root cause of the critical patch in February 2004 [8]. As with SNMP, the protocol standard was designed pre-implementation.
- TCP sequence number guessing was the result of a standards process that didn't think things through, was first noticed by Robert Morris in 1985, and was the target of a corrective RFC in 1996 [9]. We sometimes get lucky; imagine if sequence number guessing was trivially possible because of how the standard was specified.
- The Wired Equivalent Privacy (WEP) protocol was where complying with a standard ensured insecurity [10]. Unlike earlier examples, the time interval between the introduction of standards-based flaws and their exposure was short, yet WEP is still in use.
- Dan Kaminsky's DNS cache poisoning work [11] expanded earlier warnings [12], but because Kaminsky's discovery was a re-discovery, we know that it was solely the public disclosure of exploitability, not the public disclosure of vulnerability, that triggered response.
- Should a message be signed then encrypted or encrypted then signed? Writing in 2001, researcher Don Davis pointed out that "Every secure

e-mail protocol, old and new, has codified naive Sign & Encrypt as acceptable security practice: S/MIME, PKCS#7, PGP, OpenPGP, PEM, and MOSS all suffer from this flaw. Similarly, the secure document protocols PKCS#7, XML-Signature, and XML-Encryption suffer from the same flaw" [13]. Davis showed that only the protocol of sign-encrypt-sign is effective, yet this flaw is still present since the S/MIME & XML groups both ignored Davis, just as the PEM group had ignored Yvo Desmedt on the same subject, 10+ yrs before. The GPG people eventually came around.

- The IPSec's committee-driven rewrite to permit username/password authentication in IKE was implemented by vendors before the committee was even done, hence producing various serious MITM issues that would take years to stamp out. Steve Kent was one of the very few who understood the issues [14].
- Marsh Ray discovered a man-in-the-middle vulnerability in the TLS standard [15]. Vendors rallied a bit for this one, but the number of unfixable implementations is high. Ray's discovery only highlighted this aspect, that of unfixable implementations.

Common-mode failure due to common-mode operations is not limited to digital worlds and security protocols.

- The US and Soviet militaries discovered EMP (electromagnetic pulse) effects early on [16]; a 1962 US nuclear test over the Pacific took down parts of the Oahu power grid. The Soviets did something quite similar in Kazakhstan. The US remediated by shielding the hell out of military gear, but eventually moved away from copper. The Soviets continued using vacuum tubes for military communications, even in planes.
- In the late '60s, phone hackers figured out how to synthesize touch-tone-style switching and billing signals. AT&T got the Feds to pass tougher laws against stealing phone service, and AT&T changed their phone-line-based protocols to something more secure.

What to do? We know that many platforms go without updates. Would it be wise to have non-compliant servers and clients treated legally as an attractive nuisance? It is unfair, but if you don't fence your swimming pool, then drowned children are your fault. Is that a good enough solution to analogize new Internet rules?

If one interprets a standard as a kind of license, then perhaps standards should come with an expire-by date. Some attacks that are not possible in today's state of the world may become possible in the future and invalidate the design environment in which a standard was crafted. Marcus Ranum has been recommending the standard "expire-by" idea for a long time. (Perhaps standards bodies need an expire-by date as well, but that's another story for another day.)

Proposed US legislation [17] is said to permit the President to shut off the Internet during times of crisis, which would (1) be impossible and (2) detonate cascading failures. Besides, as Scott Borg points out [18], disrupting the Internet is always an offensive gesture. Nevertheless, perhaps the President should be able to mandate deprecation of specific protocols, though even then it would require effort like that for Y2K. (During the 1990s, Marcus Ranum suggested: "Re-code the Internet, recompile, reboot, and blame it on Y2K.")

Sooner or later, mayn't it be a good idea to force-deprecate, say, the backbone routability of FTP, SSH v1, or SQL? However attractive that idea might first seem, it's farfetched; something like 10% of Internet backbone traffic is not protocol-identifiable, which is of interest to the SIGINT crowd

and makes protocol filtering nonsensical, even ignoring one protocol encapsulating another.

This is not just legacy; we are busy manufacturing similar situations. Kelly Ziegler notes [19] the critical ratio between firmware-update size/frequency and the available bandwidth-to-device population, i.e., utilities who want to ship new power meters with multi-MByte firmware images reachable only by sub-10Kbit/sec bandwidth. Read carefully the rationale for doing this: "It's all conforming to industry standard protocols that have been tested and vetted." In other words, the meter population could be updated within a year or so, during which the attacker would have a clear field.

So, what is the constraint on update latency for something like the electric grid? Is a year good enough? For any situation, should you take the time-to-update as the independent variable in a risk calculation and ask whether your dependence on the underlying service is too great to tolerate the resulting cycle time? If a given cycle-time is intolerable, then you have two choices: make your cycle-time shorter or make your dependence smaller.

That may be the key point: the calculus of risk as the summation of protocol dependencies. Pursuant to his potential authority to modify the Internet, should the President say to federal agencies and critical infrastructure providers alike: "I order you to be able to continue to function in the absence of the Internet"? That would mean that some agencies and/or companies would have to keep their telephone-based call centers, keep their postal service-based payment acceptance, and/or to provide software updates via CD-ROM and not just over-the-Internet download, etc.

It's time to deprecate Jon Postel's dictum and to "be conservative in what you accept." It is time to plan, for example, for the side effect of cloud computing's making some things, such as an ASN.1 compilation, less diverse but more obscure. The more we converge on standardized solutions, the more we converge on common-mode failures. We need actionable ideas on what to do when that bites hard.

REFERENCES

[1] http://www.ietf.org/tao.html.

[2] D. Reed, "End-to-End Arguments: The Internet and Beyond," 2010 USENIX Security Symposium, August 13, 2010.

[3] J. Lewis, "Docile No More: The Tussle to Redefine the Internet," 2010 USENIX Security Symposium, August 11, 2010.

[4] Common Vulnerabilities and Exposures: http://cve.mitre.org/.

[5] J.G. Steiner, B.C. Neuman, and J.I. Schiller, "Kerberos: An Authentication Service for Open Network Systems," USENIX Winter Conference, February 1988.

[6] T. Yu, S. Hartman, and K. Raeburn, "The Perils of Unauthenticated Encryption: Kerberos Version 4," Network and Distributed Systems Security Symposium, February 2004.

[7] https://www.ee.oulu.fi/research/ouspg/PROTOS_Test-Suite_c06-snmpv1.

[8] Microsoft Security Bulletin MS04-007: http://www.microsoft.com/technet/security/bulletin/ms04-007.mspx.

[9] RFC 1948: http://www.faqs.org/rfcs/rfc1948.html.

[10] W.A. Arbaugh, N. Shankar, and Y.C.J. Wan, "Your 802.11 Wireless Network Has No Clothes," *IEEE Wireless Communications*, vol. 9, no. 6 (2002),

pp. 44–51; N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11," ACM SIGMobile, July 19, 2001.

[11] US-CERT, "Multiple DNS Implementations Vulnerable to Cache Poisoning," July 8, 2008: http://www.kb.cert.org/vuls/id/800113.

[12] S.M. Bellovin, "Using the Domain Name System for System Break-ins," USENIX UNIX Security Symposium, June 1995.

[13] D. Davis, "Defective Sign & Encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML": http://world.std.com/~dtd/sign_encrypt/sign_encrypt7.html.

[14] http://www.vpnc.org/ietf-ipsec/99.ipsec/msg01734.html; http://www.vpnc.org/ietf-ipsec/98.ipsec/msg01503.html.

[15] "Authentication Gap in TLS Renegotiation": http://extendedsubset.com/?p=8.

[16] Some history at http://www.emp.us.com/emp-radiation-from-nuclear-space.html.

[17] S. 3480, as found at http://www.opencongress.org/bill/111-s3480/show.

[18] S. Borg, "How Cyber Attacks Will Be Used in International Conflicts," 2010 USENIX Security Symposium, August 13, 2010.

[19] K. Ziegler, "Grid, PhD: Smart Grid, Cyber Security, and the Future of Keeping the Lights On," 2010 USENIX Security Symposium, August 13, 2010.

MATTHEW HICKS, MURPH FINNICUM, SAMUEL T. KING, MILO M.K. MARTIN, AND JONATHAN M. SMITH

overcoming an untrusted computing base: detecting and removing malicious hardware automatically

Matthew Hicks is a fifth-year graduate student in computer science at the University of Illinois. His current research focuses on the border between hardware and system software, with a focus on FPGAs and specialized operating systems.

mdhicks2@illinois.edu

Murph Finnicum is a graduate student in computer science at the University of Illinois, where he also completed his undergraduate studies in computer engineering. His current research includes work on ambiguity-tolerant automatic programming and novel techniques to make Web browsers faster.

mfinnic2@illinois.edu

Sam King is an assistant professor in the Computer Science Department at the University of Illinois. His primary research interests are in security, operating systems, and computer architecture.

kingst@illinois.edu

Milo Martin is an associate professor in the Computer and Information Science Department at the University of Pennsylvania. His research focuses on making computers easier to design, verify, and program. Specific projects include transactional memory, adaptive cache coherence protocols, hardware-aware verification of concurrent software, and hardware-assisted memory-safe implementations of the C programming language. Dr. Martin is a recipient of the NSF CAREER award and received a PhD from the University of Wisconsin—Madison.

milom@cis.upenn.edu

Jonathan M. Smith is the Olga and Alberico Pompa Professor of Engineering and Applied Science and a professor of computer and information science at the University of Pennsylvania. He served as a program manager at DARPA 2004–2006 and was awarded the OSD Medal for Exceptional Public Service in 2006. He is an IEEE Fellow. His current research interests range from programmable network infrastructures and cognitive radios to disinformation theory and architectures for computer augmented immune response.

jms@cis.upenn.edu

THE COMPUTER SYSTEMS SECURITY arms race between attackers and defenders has largely taken place in the domain of software systems, but as hardware complexity and design processes have evolved, novel and potent hardware-based security threats are now possible. This article presents Unused Circuit Identification (UCI), an approach for detecting suspicious circuits during design time, and BlueChip, a hybrid hardware/software approach to detaching suspicious circuits and making up for UCI classifier errors during runtime.

Modern hardware design processes in many ways resemble the software design process. Hardware designs consist of millions of lines of code and often leverage libraries, toolkits, and components from multiple vendors. These designs are then "compiled" (synthesized) for fabrication. As with software, the growing complexity of hardware designs creates opportunities for hardware to become a vehicle for malice. Recent work has demonstrated that small malicious modifications to a hardware-level design can compromise the security of the entire computing system [11].

Malicious hardware has two key properties that make it even more damaging than malicious software. First, hardware presents a more persistent attack vector. Whereas software vulnerabilities can be fixed via software update patches or reimaging, fixing well-crafted hardware-level vulnerabilities would likely require physically replacing the compromised hardware components. A hardware recall similar to Intel's Pentium FDIV bug (which cost $500 million to recall five million chips) has been estimated to cost many billions of dollars today [3]. Furthermore, the skill required to replace hardware and the rise of deeply embedded systems ensure that vulnerable systems will remain in active use after the discovery of the vulnerability. Second, hardware is the lowest layer in the computer system, providing malicious hardware with control over the software running above. This low-level control enables sophisticated and stealthy attacks aimed at evading software-based defenses.

Such an attack might use a special, or unlikely, event to trigger deeply buried malicious logic that was inserted during design time. For example, attackers might introduce a circuit that detects a certain sequence of bytes into the hardware that activates the malicious logic. This logic might

escalate privileges, turn off access control checks, or execute arbitrary instructions, providing a path for the malefactor to take control of the machine. The malicious hardware thus provides a *foothold* for subsequent system-level attacks.

During the design phase, UCI flags as suspicious any unused circuitry (any circuit not activated by any of the many design verification tests). BlueChip disconnects these suspicious circuits from the rest of the trusted circuit. However, these seemingly suspicious circuits might actually be part of a legitimate circuit within the design, so BlueChip inserts circuitry to raise an exception whenever one of these suspicious circuits would have been activated. The BlueChip exception handler software is responsible for emulating the overall behavior of the hardware to allow the system to make forward progress. BlueChip's overall goal is to push the complexity of coping with malicious hardware up to a higher, more flexible and adaptable layer in the system stack.

Motivation and Attack Model

This article focuses on the problem of malicious circuits introduced during the hardware design process. Today's complicated hardware designs are increasingly vulnerable to the undetected insertion of malicious circuitry to create a hardware trojan horse. In other domains, examples of this general type of intentional insertion of malicious functionality include compromises of software development tools [14], system designers inserting malicious source code intentionally [4, 9, 10], compromised servers that host modified source code [5, 6], and products that come pre-installed with malware [1, 2, 13]. Such attacks introduce little risk of punishment, because the complexity of modern systems and prevalence of unintentional bugs makes it difficult to prove malice or to correctly attribute the problem to its source [15].

More specifically, our threat model is that a rogue designer covertly adds trojan circuits to a hardware design. We focus on two possible scenarios for such rogue insertion. First, one or more disgruntled employees at a hardware design company surreptitiously and intentionally insert malicious circuits into a design prior to final design validation with the hope that the changes will evade detection. The malicious hardware demonstrated in previous work [11] supports the plausibility of this scenario, in that small and localized changes (e.g., tens of lines in a single hardware source file) are sufficient for creating powerful malicious circuits designed for bootstrapping larger system-level attacks. We call such malicious circuits *footholds*, and such footholds persist even after malicious software has been discovered and removed, giving attackers a permanent vector into a compromised system.

The second scenario is enabled by the trend toward "softcores" and other pre-designed hardware IP (intellectual property) blocks. Many system-on-chip (SoC) designs aggregate subcomponents from existing commercial or open-source IP. Although generally trusted, these third-party IP blocks may not be trustworthy. In this scenario, an attacker can create new IP or modify existing IP blocks to add malicious circuits. The attacker then distributes or licenses the IP in the hope that some SoC creator will incorporate it and include it in a fabricated chip. Although the SoC creator will likely perform significant design verification focused on finding design bugs, traditional black-box design verification is unlikely to reveal malicious hardware.

In either scenario, the attacker's motivation could be financial or general malice. If the design modification remains undetected by final design

validation and verification, the malicious circuitry will be present in the manufactured hardware that is shipped to customers and integrated into computing systems. The attacker has achieved this without the resources necessary to actually fabricate a chip or attack the manufacturing or distribution supply chain. We assume that only one or a few individuals act maliciously (i.e., not the entire design team) and that these individuals are unable to compromise the final end-to-end design verification and validation process, which is typically performed by a distinct group of engineers.

UCI and BlueChip can be used by anyone from designers to debuggers, but the target audience is the lead designer or system integrator who advances the design to the fabrications stage. Our work assumes that this person is trustworthy and has much to lose if the hardware contains malicious circuitry. This work also relies on a testing regimen based on simulation at the hardware description level. Here the designer with signoff responsibilities can view any wire in the design, during any given cycle, and has the ability to add or remove test cases. We assume that no extra rigging is required for specialized testing (e.g., boundary scan chains); what is simulated is what will be in the fabricated chip.

The BlueChip Approach

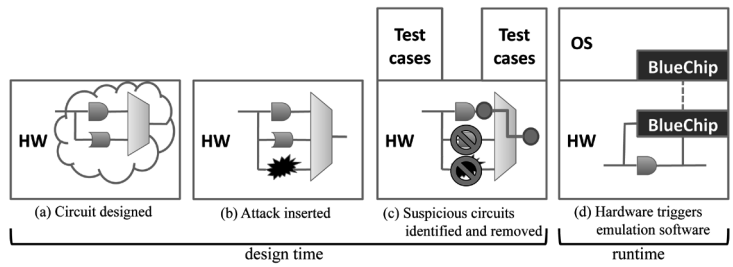


FIGURE 1: OVERALL BLUECHIP ARCHITECTURE. THIS FIGURE SHOWS THE OVERALL FLOW FOR BLUECHIP WHERE (A) DESIGNERS DEVELOP HARDWARE DESIGNS AND (B) A ROGUE DESIGNER INSERTS MALICIOUS LOGIC INTO THE DESIGN. DURING DESIGN VERIFICATION PHASE, (C) BLUECHIP IDENTIFIES AND REMOVES SUSPICIOUS CIRCUITS AND INSERTS RUNTIME HARDWARE CHECKS. (D) DURING RUNTIME, THESE HARDWARE CHECKS INVOKE SOFTWARE EXCEPTIONS TO PROVIDE THE BLUECHIP SOFTWARE AN OPPORTUNITY TO ADVANCE THE COMPUTATION BY EMULATING INSTRUCTIONS, EVEN THOUGH BLUECHIP MAY HAVE REMOVED LEGITIMATE CIRCUITS.

Our overall BlueChip architecture is shown in Figure 1. In the first phase of operation, UCI analyzes the circuit's behavior during design verification to identify candidate circuits that might be malicious. Once UCI identifies a suspect circuit, BlueChip automatically removes the circuit from the design. Because UCI might identify and BlueChip remove legitimate circuits as part of the transformation, BlueChip inserts logic to detect if the removed circuits would have been activated, and it triggers an exception if the hardware encounters this condition during runtime. The hardware delivers this exception to the software layer. The BlueChip exception handling software is responsible for recovering from the fault and advancing the computation by emulating the instruction that was executing when the exception occurred. BlueChip pushes much of the complexity up to the software layer,

allowing defenders to rapidly refine defenses, turning the permanence of the hardware attack into a disadvantage for attackers.

BlueChip can operate in spite of removed hardware because the removed circuits operate at a lower layer of abstraction than the software emulation layer responsible for recovery. *BlueChip software does* ***not*** *emulate the removed hardware directly.* Instead, it emulates the behavior of the entire hardware design using a simple, high-level, and implementation-independent specification of hardware, i.e., the processor's instruction-set-architecture specification. BlueChip software emulates the effects of the removed hardware by emulating one or more instructions, updating the processor registers and memory values, and resuming execution. The computation can generally make forward progress despite the removed hardware logic, although software emulation of instructions is slower than normal hardware execution.

In some respects our overall BlueChip system resembles floating point instruction emulation for processors that omit floating point hardware. If a processor design omits floating point unit (FPU) hardware, floating point instructions raise an exception that the OS handles. The OS can emulate the effects of the missing hardware using available integer instructions. Like FPU emulation, BlueChip uses software to emulate the effects of missing hardware using the available hardware resources. However, the hardware BlueChip removes is *not* necessarily associated with specific instructions and can trigger BlueChip exceptions at unpredictable states and events, presenting a number of challenges.

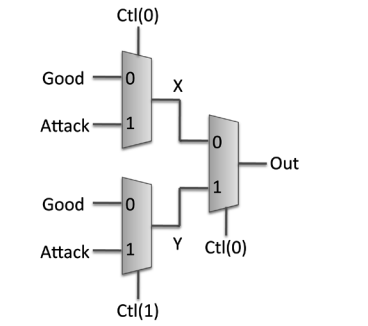
Overall, BlueChip provides a separation between the responsibilities of the hardware and the software. The BlueChip hardware prevents attacks by removing suspicious circuits. The BlueChip software ensures forward progress by emulating instructions. If an attacker is able to control the BlueChip software it does *not* give attackers any additional capabilities—the BlueChip hardware still neutralizes the attack—but usurping BlueChip software could prevent the system from making forward progress.

For more information about the design and implementation of our BlueChip hardware and software, please see our recent paper on the topic [8].

Detecting Suspicious Circuits

One key component of the overall system is the algorithm for detecting suspicious circuits. Our goal is to develop an algorithm that identifies all malicious circuits without identifying benign circuits. In addition, our technique should be impossible for an attacker to avoid, and it should identify potentially malicious code automatically without requiring the defender to develop a new set of design verification tests specifically for our new detection algorithm.

Hardware designs often include extensive design verification tests that designers and system integrators use to verify the functionality of a component. In general, test cases use a set of inputs and verify that the hardware circuit outputs the expected results. For example, test cases for processors use a sequence of instructions as the input, with the processor registers and system memory as outputs.




```
X <= (Ctl(0) = '0') ? Good : Attack
Y <= (Ctl(1) = '0') ? Good : Attack
Out <= (Ctl(0) = '0') ? X : Y
```

FIGURE 2: CIRCUIT DIAGRAM AND HDL SOURCE CODE FOR A MULTIPLEXOR (MUX) THAT CAN PASS CODE COVERAGE TESTING WITHOUT ENABLING THE ATTACK. THIS FIGURE SHOWS HOW A WELL-CRAFTED MUX CAN PASS COVERAGE TESTS WHEN THE APPROPRIATE CONTROL STATES (CTL(0) AND CTL(1)) ARE TRIGGERED DURING TESTING. CONTROL STATES 00, 01, AND 10 WILL FULLY COVER THE CIRCUIT WITHOUT TRIGGERING THE ATTACK CONDITION. GOOD, ATTACK, X, Y, AND OUT ARE ALL WIRES WITH ARBITRARY VALUES THAT ARE FREE TO CHANGE DURING SIMULATION. FOR SIMPLICITY, ASSUME GOOD AND ATTACK NEVER CARRY THE SAME VALUE. WIRES GOOD AND ATTACK ARE THE INPUTS AND WIRE OUT IS THE OUTPUT. THE WIRES LABELED CTL(X) ARE THE CONTROL LINES FOR THEIR RESPECTIVE MUXES. THE VALUE ON THE CONTROL LINE OF A MUX DETERMINES WHICH INPUT GETS ITS VALUE PASSED ALONG AS THE OUTPUT OF THE MUX. THE 0 AND 1 LABELS ON EACH MUX IN THIS FIGURE SHOW WHICH CONTROL VALUE DRIVES WHICH INPUT VALUE TO THE OUTPUT.

An attacker can easily craft circuits that yield 100% code coverage after testing, but test cases never actually trigger the attack. For example, Figure 2 shows a multiplexer (mux) circuit that has 100% code coverage without outputting the attack value. If the verification test suite includes control states (value of "Ctl(0,1)") 00, 01, and 10, all lines of code that make up the circuit will be covered, but the output value on wire "Out" will always be the same value as the value on wire "Good." We apply this evasion technique to the attacks we evaluate and find that it does evade code coverage–based detection.

Our approach is to use design verification tests to help detect malicious circuits, repurposing functional verification tests as security verification tests. If an attack circuit contaminates the output for a test case, the designer would know that the circuit is operating out-of-spec, detecting the attack. However, recent research has shown how hardware attacks can be implemented using small circuits that are designed not to trigger during routine testing [11]. This evasion technique works by guarding the attack circuit with triggering logic that enables the attack only when it observes a specific sequence of events or a specific data value (e.g., the attack triggers only when the hardware encounters a predefined 128-bit value). This attack-hiding technique works because malicious hardware designers can avoid perturbing outputs during testing by hiding deep within the vast state space of a design, but can still enable attacks in the field by inducing the trigger sequence. A processor with 16 32-bit registers, a 16k instruction cache,

a 64k data cache, and 300 pins has *at least* 2^{655872} states, and up to 2^{300} transition edges. Our proposal is to consider circuits suspicious whenever a design includes them, but the circuit does *not* affect any of the outputs for any of the test cases.

This section describes our algorithm, called unused circuit identification (UCI), for identifying potentially malicious circuits at design time. Our technique focuses on identifying portions of the circuit that do not affect outputs during testing.

```
// step one: generate data-flow graph
// and find connected pairs
pairs = {connected data-flow pairs}

// step two: simulate and try to find
// any logic that does not affect the
// data-flow pairs for each simulation clock cycle
   for each pair in pairs
      if the sink and source not equal
        remove the pair from the pairs set
```

FIGURE 3: IDENTIFYING POTENTIALLY MALICIOUS CIRCUITS USING OUR UCI ALGORITHM

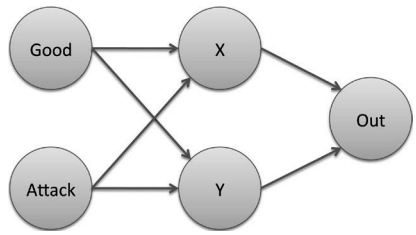


FIGURE 4: DATA-FLOW GRAPH FOR MUX REPLACEMENT CIRCUIT

To identify potentially malicious circuits, our algorithm performs two steps (Figure 3). First, UCI creates a data-flow graph for the design (Figure 4). In this graph, nodes are signals (wires) and state elements (to account for data flow across clock cycles); edges indicate data flow between the nodes. Based on this data-flow graph, UCI generates a list of all signal pairs, or *data-flow pairs*, where data flows from a source signal to a sink signal. This list of data-flow pairs includes both direct dependencies (e.g., (Good, X) in Figure 4) and indirect dependencies (e.g., (Good, Out) in Figure 4). For more details on the third member of data-flow tuples, refer to our recent paper [8]. Each data-flow tuple effectively states that all the logic between the source and the sink wire can be replaced by a short-circuit-like delay line.

Second, UCI simulates the HDL code using standard design verification tests to find the set of data-flow tuples where intermediate logic does *not* affect the data value that flows between the source and sink wires. To test for this condition, at each simulation step UCI checks for inequality for each of the remaining data-flow tuples. If the current value of the sink is not equal to the value of the source DELAY cycles ago, this implies, conservatively, that the logic between the two wires has an effect on the value. UCI removes this tuple from suspicion, as there is now a case where the intermediate logic had an effect and the effect was verified within the design's specification. More clearly, for registers, UCI accounts for latched data by maintaining a history of simulation values, allowing it make the appropriate comparison of source and sink wire values when they are separated by state elements.

After the simulation completes, UCI has a set of remaining data-flow tuples where the logic in between the source and sink wires of the tuple does not affect the value as it travels, possibly across clock cycles, from source to

sink. In other words, we could replace the intermediate logic with a wire, possibly including some delay state elements, and it would not affect the overall behavior of the circuit, in any way, for any of the design verification tests.

Consider how this algorithm works for the mux-replacement circuit shown in Figure 2, when the attack lies dormant during the test cases (values of "Attack" and "Out" are not equal):

- UCI creates the initial set of data-flow tuples, (Good, X, 0), (Attack, X, 0), (Good, Y, 0), (Attack, Y, 0), (Good, Out, 0), (Attack, Out, 0), (X, Out, 0), and (Y, Out, 0).
- UCI considers the first simulation step where Ctl(0) and Ctl(1) are 0. Thus, X, Y, and Out all carry the same value as Good. UCI removes tuples (Attack, X, 0), (Attack, Y, 0), and (Attack, Out, 0), since X, Y, and Out don't carry the same value as Attack.
- UCI considers the second simulation step where Ctl(0) is 0 and Ctl(1) is 1. Thus, X and Out carry the same value as Good, while Y carries the same value as Attack. UCI removes tuples (Good, Y, 0) and (Y, Out, 0), since Y doesn't carry the same value as Good or Out.
- UCI considers the third simulation step where Ctl(0) is 1 and Ctl(1) is 0. Thus, X carries the same value as Attack while Y and Out carry the same value as Good. UCI removes tuples (Good, X, 0) and (X, Out, 0) since X doesn't carry the same values as Good or Out.
- UCI finishes the simulation, and the only remaining tuple is (Good, Out, 0). This tells the designer that the intermediate logic between wires Good and Out doesn't have any real effect for the test cases. This implies that wires Good and Out can be short-circuited with no adverse effects.

The resulting output from UCI for this example identifies the malicious circuit without identifying any additional signals. Because it systematically identifies circuits that avoid affecting outputs during testing, BlueChip connects the wire labeled "Good" directly to the wire labeled "Out," thus removing the malicious logic from the design.

BlueChip Evaluation

In our evaluation, we measure (1) BlueChip's ability to stop attacks, (2) BlueChip's ability to successfully emulate instructions that used hardware removed by BlueChip, and (3) the runtime overhead of our system.

We based our hardware implementation on the Leon3 processor [7] design. Our prototype is fully synthesizable and runs on an FPGA development board that includes a Virtex 5 FPGA, CompactFlash, Ethernet, USB, VGA, PS/2, and RS-232 ports. The Leon3 processor implements the SPARC v8 instruction set [12], and our configuration uses eight register windows, a 16KB instruction cache, and a 64KB data cache, includes an MMU, and runs at 100MHz, which is the maximum clock rate we are able to achieve for the unmodified Leon3 design for our target FPGA. For the software, we use a SPARC port of the Linux 2.6.21.1 kernel on our FPGA board, and we install a full Slackware distribution on our system. By evaluating BlueChip on an FPGA development board and by using commodity software, we have a realistic environment for evaluating our hardware modifications and accompanying software systems.

To evaluate BlueChip's ability to prevent and recover from attacks, we wrote software that activates the malicious hardware described in our recent papers [11, 8]. Our prior work on developing hardware attacks focused on adding minimal additional logic gates as a *foothold* for a system-level attack. We explored three such footholds: the *supervisor transition* foothold

enables an attacker to transition the processor into supervisor mode to escalate the privileges of user-mode code, the *memory redirection* foothold enables an attacker to read and write arbitrary virtual memory locations, and the *shadow mode* foothold enables an attacker to pass control to invisible firmware located within the processor and take control of the system. Previous work has shown how these types of footholds can be used as part of a system-level attack to carry out high-level, high-value attacks, such as escalating privileges of a process or enabling attackers to log in to a remote system automatically [11].

To identify suspicious circuits, we used the Gaisler test suite that comes bundled with the Leon3 hardware's HDL code, the official SPARC verification tests from SPARC International, and a few custom test cases we wrote for this experiment.

To measure execution overhead, we used three workloads that stressed different parts of the system: `wget` fetches an HTML document from the Web and represents a network bound workload, `make` compiles portions of the `ntpdate` application and stresses the interaction between kernel and user modes, and `djpeg` decompresses a 1MB jpeg image as a representative of a compute-bound workload. To address variability in the measurements, reported execution time results are the average of 100 executions of each workload relative to an uninstrumented base hardware configuration. All of our overhead experiments have a 95% confidence interval of less than 1% of the average execution time.

DOES BLUECHIP PREVENT THE ATTACKS?

There are two goals for BlueChip when aiming to defend against malicious hardware. The first and most important goal is to prevent attacks from influencing the state of the system. The second goal is for the system to recover, allowing non-malicious programs to make progress after an attempted attack.

Attack	*Prevent*	*Recover*
Privilege Escalation	√	√
Memory Redirection	√	
Shadow Mode	√	√

FIGURE 5: BLUECHIP ATTACK PREVENTION AND RECOVERY

The results in Figure 5 show that BlueChip successfully prevents all three attacks, meeting the primary goal for success. BlueChip meets the secondary goal of recovery for two of the three attacks, but it fails to recover from attempted activations of the memory redirection attack. In this case, the attack is prevented, but software emulation is unable to make forward progress. Upon further examination, we found that the attack stored state that fell outside of our BlueChip hardware mechanisms. We believe that this limitation is an artifact of our current implementation and could be fixed by using a more sophisticated hardware analysis algorithm.

IS SOFTWARE EMULATION SUCCESSFUL?

BlueChip justifies its aggressive identification and removal of suspicious circuits by relying on software to emulate any mistakenly removed functionality. Thus, BlueChip will trigger spurious exceptions (i.e., those exceptions that result from removal of logic mistakenly identified as malicious). In our experiments, all of the benchmarks execute correctly,

indicating that BlueChip correctly recovers from the spurious BlueChip exceptions that occurred in these workloads.

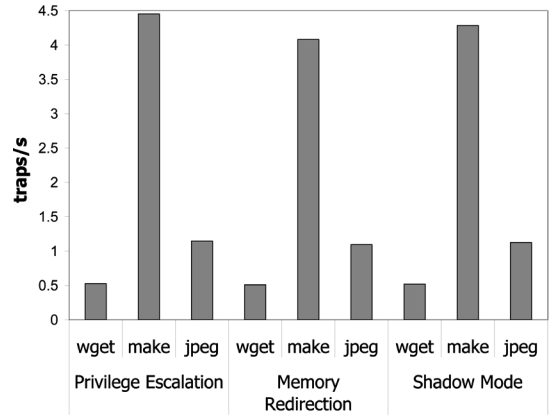


FIGURE 6: BLUECHIP SOFTWARE INVOCATION FREQUENCIES

Figure 6 shows the average rate of BlueChip exceptions for each benchmark. Even in the worst case, where a BlueChip exception occurs every 200ms on average, the frequency is far less than the operating system's timer interrupt frequency. The rate of BlueChip exceptions is low enough to allow for complex software handlers without sacrificing performance.

The discrepancy in the number of traps experienced by each benchmark is worth noting. The make benchmark experiences the most traps, by almost an order of magnitude. Looking at the UCI pairs that fire during testing, and looking at the type of workload make creates, the higher rate of traps comes from interactions between user and kernel modes. This happens more often in make than the other benchmarks, as make creates a new process for each compilation. More in-depth tracing of the remaining UCI pairs reveals that many pairs surround the interaction between kernel mode and user mode. Because UCI is inherently based on design verification tests, this perhaps indicates the parts of hardware least tested in our three test suites. Conversely, the relatively small rate of BlueChip exceptions experienced by wget is due to its I/O (network) bound workload. Most of the time is spent waiting for packets, which apparently does not violate any of the UCI pairs remaining after testing.

IS BLUECHIP'S RUNTIME OVERHEAD LOW?

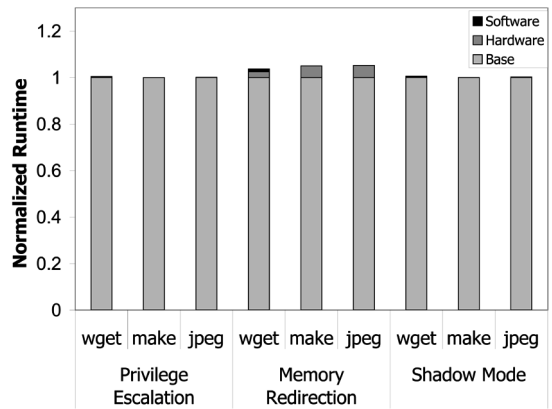


FIGURE 7: APPLICATION RUNTIME OVERHEADS FOR BLUECHIP SYSTEMS

Although BlueChip successfully executes our benchmark workloads, frequent spurious exceptions have the potential to significantly impact system performance.

Figure 7 shows the normalized breakdown of runtime overhead experienced by the benchmarks running on a BlueChipped system versus an unprotected system. The runtime overhead from the software portion of BlueChip is just 0.3% on average. The software overhead comes from handling spurious BlueChip exceptions, primarily from just two of the UCI pairs. The average overhead from the hardware portions of BlueChip, including the cases with zero hardware overhead, is approximately 1.4%.

Conclusion

BlueChip neutralizes malicious hardware introduced at design time by identifying and removing suspicious hardware during the design verification phase, while using software at runtime to emulate hardware instructions to avoid erroneously removed circuitry.

Experiments indicate that BlueChip is successful at identifying and preventing attacks while allowing non-malicious executions to make progress. Our malicious circuit identification algorithm, UCI, relies on the attempts to hide functionality to identify candidate circuits for removal. BlueChip replaces circuits identified by UCI with exception logic, which initiates a trap to software. The BlueChip software emulates instructions to detour around the removed hardware, allowing the system to attempt to make forward progress. Measurements taken with the attacks inserted show that such exceptions are infrequent when running a commodity operating system using traditional applications.

In summary, these results show that addressing the malicious insider problem for hardware design is both possible and worthwhile, and that approaches can be cost-effective and practical.

ACKNOWLEDGMENTS

The authors thank David Wagner, Cynthia Sturton, Bob Colwell, and the anonymous reviewers for comments on this work. We thank Jiri Gaisler for assistance with the Leon3 processor and Morgan Slain and Chris Larsen from SPARC International for the SPARC certification test benches.

This research was funded in part by the National Science Foundation under grants CCF-0811268, CCF-0810947, CCF-0644197, and CNS-0953014, AFOSR MURI grant FA9550-09-01-0539, ONR under N00014-09-1-0770 and N00014-07-1-0907, donations from Intel Corporation, and a grant from the Internet Services Research Center (ISRC) of Microsoft Research. Any opinions, findings, and conclusions or recommendations expressed in this paper are solely those of the authors.

REFERENCES

[1] Maxtor basics personal storage 3200: http://seagate.custkb.com/seagate/crm/selfservice/search.jsp?DocId=205131&NewLang=en.

[2] Apple Computer Inc., Small Number of Video iPods Shipped with Windows Virus. 2006: http://www.apple.com/support/windowsvirus/.

[3] Bob Colwell, personal communication, March 2009.

[4] BugTraq Mailing List, Irssi 0.8.4 backdoor, May 2002: http://archives.neohapsis.com/archives/bugtraq/2002-05/0222.html.

[5] CERT Advisory CA-2002-24 Trojan Horse OpenSSH Distribution, technical report, CERT Coordination Center, 2002: http://www.cert.org/advisories/CA-2002-24.html.

[6] CERT Advisory CA -2002-28 Trojan Horse Sendmail Distribution, technical report, CERT Coordination Center, 2002: http://www.cert.org/advisories/CA-2002-28.html.

[7] Gaisler Research, Leon3 synthesizable processor: http://www.gaisler.com/cms/index.php?option=com_content&task=view&id=13&Itemid=53.

[8] M. Hicks, M. Finnicum, S.T. King, M.M.K. Martin, and J.M. Smith, "Overcoming An Untrusted Computing Base: Detecting and Removing Malicious Hardware Automatically," in *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, May 2010.

[9] P.A. Karger and R.R. Schell, "Multics Security Evaluation: Vulnerability Analysis," technical report ESD-TR-74-192, HQ Electronic Systems Division: Hanscom AFB, June 1974.

[10] P.A. Karger and R.R. Schell, "Thirty Years Later: Lessons from the Multics Security Evaluation," in *ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference* (IEEE Computer Society, 2002), p. 119.

[11] S.T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, "Designing and Implementing Malicious Hardware," in *Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '08)*, April 2008.

[12] SPARC International Inc., SPARC v8 processor: http://www.sparc.org.

[13] B. Sullivan, "Digital Picture Frames Infected with Virus," January 2008: http://redtape.msnbc.com/2008/01/digital-picture.html.

[14] K. Thompson, "Reflections on Trusting Trust," *Communications of the ACM*, vol. 27, no. 8, 1984, pp. 761–63.

[15] United States Department of Defense, "Mission Impact of Foreign Influence on DoD Software," September 2007.

DAVID N. BLANK-EDELMAN

practical Perl tools: family man

David N. Blank-Edelman is the director of technology at the Northeastern University College of Computer and Information Science and the author of the O'Reilly book *Automating System Administration with Perl* (the second edition of the Otter book), available at purveyors of fine dead trees everywhere. He has spent the past 24+ years as a system/network administrator in large multi-platform environments, including Brandeis University, Cambridge Technology Group, and the MIT Media Laboratory. He was the program chair of the LISA '05 conference and one of the LISA '06 Invited Talks co-chairs. David is honored to have been the recipient of the 2009 SAGE Outstanding Achievement Award and to serve on the USENIX Board of Directors beginning in June of 2010.

dnb@ccs.neu.edu

I CAN'T TELL WHETHER I SHOULD apologize more profusely for writing another column about a family of Perl modules or for writing one inspired, at least in name, by a Hall and Oates song. Truth is, I'm partial to both, so I guess you'll just have to hang in there and see which bothers you the least. If it helps any, I'd recommend you check the Chromeo episode at http://www.livefromdarylshouse.com/currentep.html?ep_id=35 for some splendid updates of a few classic hits (including this column's title).

But you are in luck! In this column we're going to be discussing a family of modules that are bound together, not only in name, but also by their exceptional usefulness. We're going to explore a number of modules that modestly call themselves Something::Util(s) when, really, they should be called Something::ModulesYouAreSurprisedYouCouldLiveWithout. Some of the modules we're going to look at aren't shipped with Perl (i.e., "in the core"), but there has certainly been much discussion over the years about including them.

Working with Lists

Let's start with a core module: List::Util has been in core for eons, but I still encounter people who have never heard of it. I know I came to it relatively late, so perhaps a little reminder of how useful it is will be in order.

List::Util can export the following subroutines for use in your code:

- first—returns the first instance of something in a list
- max/maxstr—returns the element with the highest numerical/string value
- min/minstr—returns the element with the lowest numerical/string value
- reduce—reduces a list (more on this in a second)
- shuffle—returns the elements of a list in random order
- sum—adds up all of the elements in a list

Two of these deserve a little more exploration. The first on the list, first(), comes in handy in a common case that may not be obvious. Many is the time when I've found myself wanting to know if a particular string can be found in any of the elements of a list. For people with a UNIX

background, the reptile part of our brain that holds UNIX's semi-obtuse command names jumps in and suggests using grep(), as in:

```
if (grep {/fred/} @list ) { do_something; };
```

The problem with this is grep(), like its command counterpart, wants to search the entirety of its target. If you type "grep fred file" it will search all of file and show you all of the matches. Likewise, no matter how many terabytes of memory long @list is above, grep() will search all of it before returning a result. My task was "are there any?" not "what are all?" matches, so grep() is potentially doing a lot more work than we need. When you point this out to a new programmer, their next shot is a for() loop that contains an if-block with a test and a separate statement designed to exit the loop when it finds the first element that matches. That works, but it is too much code. A more concise version:

```
use List::Util qw(first);

if (first {/fred/} @list ) { do_something; };
```

The second function that deserves greater mention is reduce(). This function (more precisely, the abstract notion of a function with this definition) has been very popular in the trade press and among the big data crowd because it is the second part of the much vaunted MapReduce approach. Here's how the doc describes it:

> Reduces LIST by calling BLOCK, in a scalar context, multiple times, setting $a and $b each time. The first call will be with $a and $b set to the first two elements of the list, subsequent calls will be done by setting $a to the result of the previous call and $b to the next element in the list.

So let's look at a quick example to make this description a little clearer:

```
use List::Util qw(reduce);
print reduce { $a * $b } (1,2,3,4,5); # prints 120
```

How did we get 120?

The reduce() call starts by assigning 1 to $a and 2 to $b. It multiplies them, assigns the result of 2 to $a (1 times 2 is 2) and 3 (the next number in the list) to $b. It multiplies 2 times 3 yielding 6.

This process repeats:

- assigning 6 to $a and 4 to $b, multiplying 6 times 4 to get 24
- assigning 24 to $a and 5 to $b, multiplying 24 times 5 to get 120

The end result is we've reduced the initial list of five elements down to a single result. We're using a very simple reduction operation (multiplication), but that initial block can contain code that manipulates $a and $b in any way you'd like.

List::Util looks like a handy collection of list manipulation functions, but they are all pretty basic. If we are willing to step out of core, we will be welcomed with open arms by List::MoreUtils. List::MoreUtils has quite a few more Utils. Most are of the form:

```
function_name {some piece of code} @list;
```

Let me break them down into some rough categories for you.

- list element membership questions: any, all, none, notall

  This allows you to write things like:

```
use List::MoreUtils qw(any none);
if (any {/fred/} @list) { do_something };
if (none {/fred/} @list) { do_something };
```

For consistency's sake, I used /fred/ in the code blocks above, but really, that could be any code that makes a decision by returning true or false when presented with each element of the list (via $_). Instead of a basic regexp test, we could use anything:

```
if (any {$_ == 3} @list) { do_something };
if (any {people_want_to_come_out_and_play($_)}) { do_something; };
```

- list member counts: true, false

  These return the number of elements for which the {code} block is true or false, as in:

  ```
  use List::MoreUtils qw(true false);
  my $num_of_fredful_items      = true {/fred/} @list;
  my $num_of_items_lacking_fred = false {/fred/} @list;
  ```

- list member searches: firstidx (first_index), lastidx (last_index), firstval (first_value), lastval (last_value), uniq

  The first set of functions in this category (and their aliases, listed in parens above) will locate the first or the last occurrence in the list where their {code} block becomes true and returns either the index of or the actual value at that spot. The uniq() function will either return a list of the uniq elements in your list or the number of unique elements, depending on how you call it.

- list insertion: insert_after, insert_after_string

  This really nifty function makes for more readable code than the usual push/pop/shift kind of dance used when you want to insert an element into the middle of a list. You tell it a way to locate an element, either by {code} or a simple string, and it will insert the value of your choice into the string right after the first item it finds. For example:

  ```
  use List::MoreUtils qw(insert_after_string);
  my @list = ('sing','a','song');
  insert_after_string 'a','joyous', @list # @list now has sing,a,joyous,song
  ```

- list splitters: before, after, part

  These functions make it easy to split a list into two or more sub-lists. The before() and after() functions will return the parts of a list either before or after the first place its {code} argument is true. The part() function is a bit more sophisticated. It can partition a list into N buckets (sub-lists) based on its {code} argument, returning a list of references pointing to each sub-list. Here's an example adapted from the List::MoreUtils test suite:

  ```
  use List::MoreUtils qw(part);
  my @list = 1 .. 12;
  my $i = 0;
  my @part = part { $i++ % 3 } @list;
  ```

  It yields a list with references to the following sub-lists:

  ```
  [ 1, 4, 7, 10 ]
  [ 2, 5, 8, 11 ]
  [ 3, 6, 9, 12 ]
  ```

  I had to stare at this for a few seconds before I understood what was going on, so let me tell you how this works. It is a little confusing because up until now in this column, {code} always contained a reference to the current element. In this example, we're going to not bother to look at the value of each element and instead will be using a simple counter to calculate where things go.

We create a list whose values are the numbers 1 through 12. We then initialize a counter called $i. The part() function iterates over each element in our list. For each element in the list, we divide the counter value (what is in $i) by 3 and check the remainder. If the remainder is 0, the value of the list at that point will be placed in the first sub-list; if the remainder is 1, it goes in the second sub-list; if the remainder is 2, into the third sub-list, and so on. It goes something like this:

$i = 0, 0 % 3 = 0, so the first element of @list (1) is placed into the first sub-list
$i = 1, 1 % 3 = 1, so the second element of @list (2) is placed into the second sub-list
$i = 2, 2 % 3 = 2, so the third element of @list (3) is placed into the third sub-list
$i = 3, 3 % 3 = 0, so the fourth element of @list (4) is placed into the first sub-list...

- Tweaks: apply, indexes, minmax()

  The first two are small twists on the standard Perl functions of map() and grep(). apply() behaves like map(), except it makes sure you can't perturb the list during the iteration in the same way you can with map(). With map(), if you write:

  ```
  my @results = map {$_ += 5} @list;
  ```

  both @results and @list contain all of the elements of @list with 5 added to them. Some people like to write code that changes a list using some sort of map()'d assignment (not a great idea), and others do it by mistake. If we ran the same code with apply() instead of map(), @list would remain pristine after the operation was over.

  The twist on grep() called indexes() takes the same arguments as grep(), but instead of returning a list of the values found that match {code}, it returns the list of indices where {code} evaluated to true.

  minmax() is more of a tweak on the min() and max() functions we saw in List::Util. It returns a single two-element list with both the minimum and the maximum values found on a given list.

- multi-list functions: mesh (zip), each_array/each_arrayref, pairwise

  This is the final category of functions in List::MoreUtils. These functions let you operate on multiple lists at once. For example, the mesh() function (or its alias, zip()) takes N lists and returns a single list created by collecting the Nth element of each list, followed by the N+1th element, etc. The resulting list consists of the first elements from each of the lists, followed by those elements that were in the second position in each list, followed by those in the third position, and so on. As an aside, if the idea of a zip() operator in Perl sounds vaguely familiar to you, that's because you used your time machine to go into the future where you spied a fairly sophisticated version implemented in Perl 6.

  If you dig how mesh()/zip() work, you'll probably also like each_array() and its variant each_arrayref(). Both of these functions work by creating an iterator for you that will hand back the Nth item from each of those lists. The pairwise() function lets you operate on two lists, a pair of items at a time, using the same {code} block.

  I realize we've been talking about list-related Utils for quite some time, so let me mention one more thing as an outro. It is hard to tell just how tongue-and-cheek this is, but Dave Rolsky says in his List::AllUtils doc: "Are you sick of trying to remember whether a particular helper is

defined in List::Util or List::MoreUtils? I sure am. Now you don't have to remember. This module will export all of the functions that either of those two modules defines."

I'm not clear this is a big problem, but hey . . .

Working with Hashes

By now you are probably sick of list manipulation functions. We'll switch to manipulating a different data type to cleanse your palette. For this section, I want to look at another module found in the Perl core (as of Perl 5.8): Hash::Util. I can't say I use it nearly as often as the previous modules, but it has some functionality that can improve the safety of your code immeasurably under the right circumstances.

Hash::Util has a number of functions that allow you to lock the contents of a hash in different ways. Devoted readers of this column may remember the two-part series on tie()-based modules I did back in August and October of 2006. In that series I mentioned Tie::StrictHash. The functions in this module can operate much like that module, only without all of the funny tie() business. Plus, you get three different granular levels of "locked."

The functions lock_hash() and unlock_hash() operate as you would expect. Once locked, nothing in the hash, either keys or values, can be changed. Any attempt to change anything in the hash produces an error like:

```
Modification of a read-only value attempted at...
```

The next level of locked is lock_keys() and unlock_keys(). If you just call on a hash the way you would on lock_hash(), it will stop the addition of keys to the hash with a message like:

```
Attempt to access disallowed key 'johnnycomelately' in a restricted hash at...
```

Any keys already in the hash when you locked it will be considered "allowed." If you'd like to be more specific about which keys should be allowed besides "anything in the hash at the time of locking," lock_keys can also take a second argument of a list of allowed keys. You can test for presence in a locked hash using exists() as per usual, and can even delete entries using delete(). If you delete a key, you are allowed to put it back into the hash because that key name stays on the "allowed" list.

When you've locked the keys in a hash, you can change the values of any of the allowed keys, which leads us to the final level of granularity: the functions lock_value() and unlock_value() (saw this coming, right?). Code like this:

```
use Hash::Util qw(lock_value);
my %hash = ('dollar' => 1, 'euro' => 1.2, 'yen' => 3);
lock_value(%hash, 'dollar');
lock_value(%hash, 'euro');
lock_value(%hash, 'yen');
```

will make sure that the values for those keys in the hash can't be changed. The ability to lock hashes like this means you can avoid writing code that either puts the wrong keys or the wrong values into a critical array in your program (e.g., with a typo).

Before we switch to the last gear, I should mention that there are a couple of modules worth checking out that can do some nifty things with hashes but that I won't have time to cover here: Deep::Hash::Util offers some easy ways to work with nested hashes, and List::Pairwise (I know, it doesn't end in

::Util, but perhaps it should) lets you do all sorts of neat things with pairs of elements both in lists and in hashes.

Working with Strings

Old hands at Perl (aye, you scurvy dogs) are probably expecting the final section to discuss utilities associated with scalars, the last remaining major data type in Perl. I would do that except the obvious choice of module for this would be Scalar::Util. The problem is—Scalar::Util is boring, boring, boring. Unless you need to interrogate a scalar to determine if it is blessed, tainted, weak, etc., it won't do very much for you. It's worth knowing that there is a module that can provide this information, but if I had to think of the last time I used Scalar::Util in regular code, I'd probably start reminiscing about the days when computers ran on coal and had to be stoked periodically.

The closest thing to a useful module for this datatype is String::Util. It has a number of syntactic sugar functions like trim (to remove whitespace), crunch (to remove multiple occurrences of whitespace and to trim), nospace (to remove whitespace), and so on. All of those could easily be done with regular expression substitutions, but using a named function may be easier to read.

That's all of the utilities we have time for today. I'd recommend poking around on CPAN for the other modules that have ::Util in their name, because there is some neat stuff there. Take care, and I'll see you next time.

PETER BAER GALVIN

Pete's all things Sun: comparing Solaris to RedHat Enterprise and AIX

Peter Baer Galvin is the chief technologist for Corporate Technologies, a premier systems integrator and VAR (www.cptech.com). Before that, Peter was the systems manager for Brown University's Computer Science Department. He has written articles and columns for many publications and is co-author of the *Operating Systems Concepts* and *Applied Operating Systems Concepts* textbooks. As a consultant and trainer, Peter teaches tutorials and gives talks on security and system administration worldwide. Peter blogs at http://www.galvin.info and twitters as "PeterGalvin."

pbg@cptech.com

THESE ARE TRYING TIMES IN SOLARIS-land. The Oracle purchase of Sun has caused many changes both within and outside of Sun. These changes have caused some soul-searching among the Solaris faithful. Should a system administrator with strong Solaris skills stay the course, or are there other operating systems worth learning? The decision criteria and results will be different for each system administrator, but in this column I hope to provide a little input to help those going down that path.

Based on a subjective view of the industry, I opine that, apart from Solaris, there are only three worthy contenders: Red Hat Enterprise Linux (and its identical twins, such as Oracle Unbreakable Linux), AIX, and Windows Server. In this column I discuss why those are the only choices, and start comparing the UNIX variants. The next column will contain a detailed comparison of the virtualization features of the contenders, as that is a full topic unto itself.

Choices

There are certainly many operating systems, and many of those are "good." However, there are only some that a Solaris administrator would find interesting professionally. Those are the operating systems that could subsume, or at least co-exist with, Solaris in a datacenter. Such operating systems need to be scalable, reliable, secure, and powerful. For a sysadmin to bother devoting the time and intellectual effort to learning a new operating system, it must also have a future, both technologically and commercially. Based on that reasoning, many operating systems fail to make the cut. Linux certainly has a future, and meets the aforementioned datacenter qualifications. Red Hat Enterprise Linux (RHEL) is the leading commercial Linux in the USA, if not the world, so that version is included. AIX 7.1 is (surprisingly to me) feature rich, and runs on the excellent Power 7 CPU, making it a worthy contender. HP-UX is not included on my list, as its feature set has not kept pace, and the servers it runs on don't appear to be compelling enough for a site to switch from their Sun servers. Windows Server 2008 is certainly a valid choice, but a technical comparison between it and other operating systems is unlikely to cause anyone to switch toward or away from it. And

let's not forget Solaris, which in all likelihood will continue to be a leading datacenter operating system.

The Comparison

This column is about choice, and which other operating systems are worth learning by a professional Solaris system administrator. Now that the field of contenders is narrowed to RHEL, AIX, and Solaris, how can they be compared? Operating systems have many aspects, and really all of these should be considered when trying to chose the "best" OS. What follows is a set of criteria, and an attempt to compare the operating systems and their abilities within that criteria. Some criteria are certainly objective, while others are necessarily subjective. I try to gather all of the important data together, and provide the details needed for analysis. What is not included is any sense of the importance of a given criterion, as that will vary by administrator and by site. That priority is what every admin will need to add to their decision process.

There are many base features that all operating systems share, and still more that all UNIX-based operating systems have. And while the details on those may vary, the net result is that there is little generally important differentiation. Of course, a small detail in some aspect of some operating system may be vitally important for a given use case, but that is impossible to include in a comparison such as this. In the discussions below, where features are on a par at the macro-level, I call them equivalent even though there might be differences of note.

Although no Solaris overview should be needed by the readers of this column, a quick overview of the other contenders would be worthwhile.

RHEL Overview

RHEL is Red Hat's commercial Linux release. Linux, of course, runs on x86 and some other CPUs, and is commonly found on everything from embedded systems through mobile devices, desktops, and servers. It is certainly ubiquitous, and many datacenters run a mix of Windows, Linux, and their "core" operating system. RHEL requires a maintenance contract for support, and it is reasonably priced. However, some sites choose to run CentOS, the almost exact duplicate of the RHEL release but delayed somewhat and without the support program or the support expense. Of course, there are very many other Linux distributions (see http://en.wikipedia.org/wiki/Comparison_of_Linux_distributions for a full list). Fedora is much like OpenSolaris used to be—a bleeding-edge distribution full of new features not yet included in the commercial and supported release. Oracle also has Oracle Enterprise Linux, which is essentially a fork of CentOS and much like RHEL except with lower-cost support via Oracle. ISV support of the various Linux distributions varies quite a lot, so be sure to check for availability of your applications.

As a side note, many sites start with CentOS on their journey to Linux, and sooner rather than later find themselves running Linux for core production use and still using CentOS. Should there be a problem at that point, the site is self-maintaining their releases and self-debugging any problems. If your site is running Linux for important production uses, consider whether self-maintenance or commercial support is the best course of action. It would be unfortunate to be in a situation where the administrators, IT managers, or business unit managers believe a facility is fully "supported" only to find that when there is a problem only some parts are completely covered.

AIX Overview

While Linux is common, AIX is less so, and perhaps a bit of a mystery to many sites. The problem with AIX is that it has somewhat of a "weird UNIX, not very common, feature-poor" reputation. Until recently, that was certainly how I regarded it. And while it is true that AIX is slightly different from other UNIX versions and only runs on one kind of hardware, the Power CPU architecture from IBM (meaning that it is difficult to explore), AIX has evolved rapidly over the past few years.

A Power CPU-based system is needed for familiarity with AIX. Fortunately, AIX now runs on a large suite of systems, ranging from blades, through two-rack-unit (2RU) systems, and up to full-rack systems. A reasonably small investment would allow a datacenter to run AIX on Power for full testing. RHEL and SUSE, as well as AIX, are available on the Power CPU, which may be a consideration for some sites. Using Power's virtualization features, all three OSes could be run on the same hardware, but that is a tale for the next column.

AIX itself now has a rich feature set, including parity with Solaris in some areas and even excelling it in others, as described in detail in the next section.

Features

Comparing operating systems is difficult, probably thankless, and impossible to get absolutely right, as discussed above. Perhaps that's why there is a dearth of OS comparisons, even from the vendors of the operating systems. However, in this section is a rich set of criteria and a best-effort explanation of what each of the contending operating systems has to offer in those areas. This comparison is based on the current commercial release of each operating system: Solaris 10 9/10, AIX 7.1, and RHEL 5.5. Where there are multiple flavors of the operating system, the most advanced version is included.

PLATFORMS

Solaris is supported on x86 and SPARC CPUs, and has a hardware compatibility list (HCL) detailing which components are supported on x86 servers (http://www.sun.com/bigadmin/hcl/). Additionally, Dell and HP currently sell and support Solaris on some of their x86 products.

RHEL runs on x86, x86-64, Itanium, IBM Power, and IBM System Z servers. The full list of supported hardware is available at https://hardware.redhat.com/.

As stated above, AIX runs only on IBM Power CPU-based systems.

SCALABILITY AND PERFORMANCE

The vendors publish supported scalability information, in terms of the raw ability to use CPUs and memory.

Solaris—256 cores of CPU, 4TB of main memory.

RHEL—32 cores on x86, 256 cores on Itanium 2, 64 cores on AMD x86-64, 128 cores on IBM Power, and 64 cores on IBM System Z. Main memory limits include 16GB on x86, 2TB on Itanium 2, 256GB on AMD x86-64, 256GB on IBM Power, and 1.5TB on IBM System Z.

AIX—256 cores of CPU, 8TB of main memory.

Comparing the actual performance of operating systems is more of a challenge, with benchmarks possibly the most reasonable approach. However, benchmarks are designed to measure hardware performance or application performance, not that of the host operating system. And of course benchmarks are imperfect, being part art, part science, and part vendor manipulation. To delve more into performance, suggested sites include www.spec.org and www.tpc.org. The SPECjbb2005, a measure of Java performance, is particularly interesting because it is modern, still being used, has many submissions, and seems well thought out. Having a look through the results of specific vendors, platforms, and configurations may go a long way toward comparing real-world performance of many systems and give a feel for operating system performance.

FILE SYSTEMS

Solaris includes ZFS, as well as UFS and a variety of special-purpose operating systems. Subjectively, ZFS has no peers in current commercial file systems, but it is also the newest and has the shortest track record in terms of reliability, performance, and maintainability of the options.

RHEL has both ext3 and GFS file systems. Ext3 is journaled for reliability, but lacks other features such as snapshots, replication, deduplication, compression, and checksumming that are core to ZFS. GFS, or the Global File System, is a clustered file system that allows up to 16 Linux nodes to mount, read, and write the same files. As with ext3, it lacks the advanced features of ZFS, but ZFS is not a clustered file system, and a ZFS file system may only be mounted on one system at a time.

AIX features the JFS2 file system, which is similar to ext3 in that it includes journaling. It used to have compression (in JFS1) but, oddly, JFS2 does not. JFS2 includes snapshotting, even going beyond the ZFS feature by allowing "external" snapshots that can be mounted on other systems. However, there is no clone ability to make a snapshot read-write. Finally, it has encryption, which ZFS currently lacks. Features like ZFS deduplication and replication are missing. Also available with AIX is GPFS, IBM's clustered file system. Although not included in Linux, there are Linux ports of GPFS available, allowing a file system to be shared between AIX and Linux. For more on GPFS see http://www-03.ibm.com/systems/software/gpfs/.

VIRTUALIZATION

The virtualization features of these operating systems are extensive but quite variable. RHEL must do all of its virtualization in software, while Solaris and AIX can make use of specific hardware features of their platforms to provide other virtualization options. The next Pete's All Things Sun column will focus on comparing the virtualization offerings of Solaris, RHEL, and AIX.

DEBUGGING

Debugging was long a backwater of operating system features, until the Solaris introduction of DTrace moved it to the forefront. There are many claims and counterclaims about operating system debugging features and functions, mostly between the Linux and Solaris factions on sysadmin forums. I believe it is safe to say that Solaris has a very full-featured, integrated, and supported dynamic tracing and debugging system.

RHEL has SystemTap, which is now fully integrated and supported. In general, SystemTap is an improvement on strace and other Linux debugging tools but is not as comprehensive as DTrace.

AIX includes a features called ProbeVue, which looks very similar to DTrace. Once could argue that IBM ported DTrace to AIX and called it ProbeVue, but that could also lead to a debate. ProbeVue does lack the aggregation functions of DTrace, as well as providing fewer probe points. If you are familiar with DTrace you might want to look at the chapter on ProbeVue in the AIX manual (http://publib.boulder.ibm.com/infocenter/aix/v7r1/topic/com.ibm.aix.genprogc/doc/genprogc/genprogc.pdf) and at the ProbeVue QuickSheet (http://www.tablespace.net/quicksheet/vue-quicksheet.pdf) to see how close it is to DTrace.

INSTALLATION AND ADMINISTRATION

Generally, all three operating systems are designed not only to install from a set of media, but also to be bulk-installed over a network. Solaris has the JumpStart and Flash Archive methods. AIX has several facilities including "NIM" and "EZNIM," as well as procedures involving the HMC (Hardware Management Console). AIX has the nice feature of being able to create bootable backup tapes that can be used for restoration. RHEL provides a "network install" feature that works similarly to JumpStart.

SECURITY

Another area ripe for lively debate is security, and comparison is again complex. In general, all three of these operating systems include role-based access control (RBAC), which is a key to limiting users to just the privileges that they need to accomplish their goals and to avoid privilege escalation, where users or applications can gain more privileges than they are supposed to have. The US government provides security measurement criteria in the form of the "Common Criteria" guidelines and testing (http://www.commoncriteriaportal.org/). Such certification takes time and frequently applies to older versions of a given operating system. There are also many variations within the evaluation (specific hardware, for example) that make determining "what is more secure" difficult. In summary, all of these operating systems have a good rich set of security features, which, if used properly, can result in very secure deployments.

ISV SUPPORT

Perhaps the most important comparison point for a given facility is whether or not an operating system runs all of the applications that the facility requires. If an operating system cannot run a needed application, then it is not an option. All three vendors provide ISV lists, but further study beyond the list is required. For example, when each ISV releases a new version of its software, how long does it take for that version to become available on the operating system in question? Another important aspect is patch availability and delay. How long after a patch is released by an ISV on one platform does it take for release on the other platforms? For example, Oracle used to release Oracle Database patches for Solaris SPARC and RHEL first, on the same day, and follow that with the Solaris 10 x86 patch weeks or months later. Recently they changed that policy and release Solaris 10 x86 patches with the first wave as well. Delays in patch releases can leave sites vulnerable to bugs that affect reliability, performance, and security. Be sure

to check your important applications against the ISV lists, and check with those ISVs to determine how they treat a given operating system before making any moves between platforms. Another important sanity check, at least for the most important applications, is to poll the application vendors to see which platforms they recommend, or at least are commonly run on. Being the only site to run a given application on a given platform can induce a very lonely feeling.

How to Make a Switch

Even after all these considerations, there are many other factors to weigh when adding or replacing an operating system. Sysadmin knowledge is certainly in the forefront, as is the overall cost of such a project. All three of the considered vendors have programs available to help a site move to their products. Check their Web sites or with your favorite reseller for details. Training can include specific areas of interest, or comparison of the operating systems. For example, IBM has a four-day course designed for system administrators with knowledge of other UNIXes to quickly get them familiar and comfortable with the AIX way of performing the standard sysadmin tasks. Even with these aids, bringing in a new operating system or hardware platform can be a challenge. Certainly, migrating to a new platform should not be undertaken lightly and should be given due deliberation and planning.

Conclusion and Further Reading

Computing history is full of waves of operating system growth and shrinkage. Although the operating system choices seem to be narrowing these days, it's possible that once again we will have an explosion of choices. In the meantime, it seems that there are few contenders for datacenter managers to use as their core platform. Certainly Solaris, RHEL, and AIX are at the top of my list, and this column should provide a starting point for exploration and evaluation. In the next issue, the comparison concludes with a detailed look at the virtualization features of these products.

There are many resources available to aid in learning these operating systems and in comparing them to their peers. For a data sheet on the features of AIX 7.1, see ftp://public.dhe.ibm.com/common/ssi/ecm/en/pod03054usen/POD03054USEN.PDF.

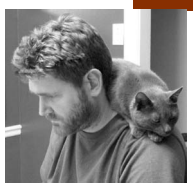
IBM has a portal containing information about migrating to Power and AIX from SPARC and Solaris: http://www-03.ibm.com/systems/migratetoibm/sun/. It also provides "Redbooks," or technical white papers, available on a wide variety of topics at http://www.redbooks.ibm.com/.

Solaris technical documents can be found at http://www.oracle.com/technetwork/server-storage/solaris/documentation/index-jsp-135724.html.

A RHEL data sheet is available at http://www.redhat.com/f/pdf/rhel-55-datasheet.pdf, while version comparison information can be found at http://redhat.com/rhel/compare.

DAVE JOSEPHSEN

iVoyeur: Ganglia on the brain



Dave Josephsen is the author of *Building a Monitoring Infrastructure with Nagios* (Prentice Hall PTR, 2007) and is senior systems engineer at DBG, Inc., where he maintains a gaggle of geographically dispersed server farms. He won LISA '04's Best Paper award for his co-authored work on spam mitigation, and he donates his spare time to the SourceMage GNU Linux Project.

dave-usenix@skeptech.org

IT'S IRONIC, I THINK, THAT I'VE BEEN dissatisfied with my RRDTool data collection and display options for as long as I have. Ironic in the sense that RRDTool is such a category killer for what it does (storing and graphing time series data). It used to be novel, to see that distinctive-looking RRDTool graph popping up somewhere unexpected, but these days it's so ubiquitous that I'm surprised to see anything but. So if RRDTool is the ultimate solution for everything that happens between data coming in and graphs going out, the open source community has surely provided category killers for polling data and displaying those graphs, yes?

No. When it comes to polling and displaying that data, we're far from having a category-killer, in my opinion. It's not for lack of candidates, mind you. Freshmeat can provide you with myriad options written in all manner of languages (but mostly PHP) that do essentially the same thing—put an RRDTool graph on a Web page. It's a bit of a cop-out to say they all do "essentially" the same thing, in fact, since they all seem to make the same mistakes over and over again. Am I missing something? Hold on a second, let me make a couple of Google searches . . .

No. I don't think I am, but please (oh please) correct me if I'm wrong. Dear people who write RRDTool polling and graphing engines, here's what I really need from you:

1. Separate your data polling from your graphing engine. Better yet, just write one or the other. You'll have to account for what other people may or may not do that way. If you're writing one tool that does them both, I should be able to use it for polling while completely ignoring your graphing component and vice versa.

2. If you're writing a polling engine, it should speak more than just SNMP, it should provide sane defaults, and it should make it easy for me to discover what they are and change them. I should be able to tell it about my SNMP data sources as well as my monitoring system agents and metric-laden log files (ideally, these should be plug-ins to a larger, more generic polling engine). I should be able to *easily* set whatever custom RRDTool attributes I want per device per metric.

3. If you're writing a graphing engine, it should be able to read round robin databases (RRDs) from any location on the hard drive, including multiple locations at once. It should be able to combine data from any number of RRDs in any number of locations. It should have the capability to automatically detect metrics within the RRDs and provide predictable URLs that allow me to quickly see a graph of any single metric in any RRD in a directory it knows about. It should also allow me to save static templates for complicated stuff. I should be able to change graphing parameters of any existing or dynamically generated graph by modifying the URL.

I'll stop there; I hadn't planned on ranting. It's just that there are so many tools out there now that come so close. It's frustrating. A few years ago (well, more like 10 now, heh) my pockets were laden with gadgetry. I had a camera, an MP3 player, a cell phone, and a RIM pager/email device. This feels a lot like that did. Can't we make one pocket-sized gadget that does all of this stuff? Please?

Cacti [1] is beautiful, but just try to use something else for polling. Drraw [2] is awesome (and I use it today), but I have to statically define every graph before I'm able to see it. General-purpose monitoring systems like Nagios can be shoe-horned into polling-engine duty by bolting on any of 1000 different Perl scripts, but a dedicated task-specific polling engine could poll more often and with a lot less overhead. Enter Ganglia [3].

Ganglia

A few years ago I was doing some load-testing work and needed a tool that could poll and display the Big Four server metrics (CPU, memory, disk, network) as close to real-time as possible and without putting any load on the systems themselves. If that last sentence didn't give you pause, then you're in a different line of work than I am. I may as well have added, "and needed a never-ending supply of $20 bills." "Well, Dave," I can hear you say, "don't we all?" Anyway, I don't remember how, but I came across Ganglia, which at the time was a somewhat nascent project. The PHP-based front end didn't work, but the polling engine looked like it did, so I looked at the PHP and made a couple of simple fixes that, if I recall correctly, had something to do with how it was generating links (I probably caught their CVS server on a bad day). Once I got it working, the results were amazing.

It put zero overhead on the servers, collected every available statistic related to the Big Four plus a bunch of platform-specific statistics, and had a polling interval short enough that you could literally see data scrolling by in the RRDTool graphs when you set the page reload to 1 second. Had it been a more mature project at the time (it was in that stage where all it needed to guarantee its demise was my unconditional devotion), I would have immediately abandoned all of my metric collection efforts in its favor. A few months ago, I had the same sort of requirement, and went back to check on its progress. To my delight I found a healthy project with a goodly sized user base, forums, mailing lists, regular releases, the works. I'm currently in the process of abandoning all of my metric collection efforts in its favor. Well, at least the ones on general-purpose operating systems where Ganglia can be installed, sigh.

To be clear, it's not the category-killer I'm looking for. In fact, the frustration permeating my muse in the paragraphs above has a lot to do with what Ganglia isn't. It has, however, allowed me to consolidate a vast amount of my metric-gathering hodge-podge into a neat, superbly scalable, and easily configured set of RRDs, and for that I am grateful. Further, Ganglia never set out to be a general-purpose RRDTool front end. It was written with a very

specific purpose in mind, so some of what it isn't is by design. For that it obviously cannot be blamed—it's a darn good design. Other things Ganglia isn't are probably pretty easy fixes that I'm sure will be patched away before long. For now, I'm going to focus on what it *is*, as well as the usual details about what my particular implementation looks like.

In the next issue, I plan to do a follow-up article to talk about Ganglia's plug-in architecture. Newer versions of Ganglia (like so many of my favorite tools) can be expanded with user-contributed modules, written in C or Python. If you read this column with any regularity, you know how I love my C-based shared-object plug-in architectures. It should come as no surprise, then, that I've written a couple of plug-ins for Ganglia in C, and can't wait to walk you through them in the next article (I know, lucky you!).

Ganglia is, as its sourceforge page says, a distributed monitoring system for high-performance computing systems such as clusters and grids, but that doesn't preclude its use for "normal" server systems. It consists primarily of two daemons and a Web front end written in PHP. The two daemons are called gmond and gmetad. Gmond may be thought of as the monitoring agent. It's installed on the systems you want to collect statistics from. Each gmond node will multicast its metrics to its peers at user-configurable intervals. Every node in a Ganglia cluster knows the status of the entire cluster.

Gmetad nodes collect statistics data from the machines running gmond. Since every gmond node knows the state of all its peers, gmetad only needs to talk to a single node in the cluster. Both daemons are written in C and make heavy use of the Apache Portable Runtime Library for portability. In my experience, they're lightweight in the extreme, and their use of multicast and node status consolidation does a great job of minimizing network chatter. Here are a few other things Ganglia gets right:

- Once a cluster is configured, new nodes are automatically detected. No configuration required.
- Automatic fail-over: A gmetad daemon can be pointed at every node in the cluster, and it will only attempt to contact the first configured node in each cluster. If that node dies, gmetad will try the next one.
- Automatic graphing: Ganglia's front end includes a graph.php program that will generate a graph of any metric for an entire cluster or an individual box. I can change its parameters by modifying the URL, so I can easily link to these graphs from anything that can take a URL. I can't change every RRDGraph parameter I'd like, but it's a great start.
- You can use as many gmetads as you'd like, and you can even have gmetads report up to other gmetads, so Ganglia Grids can scale beyond subnet boundaries and into the thousands of hosts.
- The Ganglia front end dynamically reorders all of its output based on the most utilized clusters and cluster nodes. It also does really nice things such as recoloring the backgrounds of the graphs based on user-provided thresholds. It's an interface I think Tufte [4] would generally approve of.

The Ganglia "Grid" is the sum total of every host you're monitoring with Ganglia. The Grid is composed of a number of "clusters," each of which is further composed of a number of individual servers, or nodes. Servers that make up a cluster are simply hosts that you've configured to use the same multicast address. The gmond.conf file allows you to name the cluster and provide other details such as latlog, and a description, but the configuration parameter that actually segments cluster nodes is the multicast address you assign them.

I make that point because it confused me initially. I was configuring nodes with different cluster names in the gmond.conf file, but some were

appearing in the correct clusters and others weren't. It turned out that the Cisco switches in my environment weren't properly configured for multicast, so every node in the same subnet using the same multicast address was being grouped into a cluster by Ganglia's front end, and since gmetad only talks to a single node in each cluster for the state of the entire cluster, whichever node gmetad chooses to talk to gets to actually name the cluster.

At the moment, I'm content to run a separate Ganglia Grid per data center. Since we already have a central monitoring host per data center anyway, this works out well. The Ganglia Grid in my architecture encompasses several network subnets. You don't have to deal with multicast forwarding for this to work as long as nodes in the same cluster don't bypass a subnet boundary. Even if you do have two nodes in the same cluster that are in different network subnets, you can bypass the multicast networking headaches by configuring those hosts to update each other via unicast instead. This is easily done: sample syntax resides in the gmond.conf sample configuration file, which can be generated by running gmond with a -t switch. We use several openBSD systems in our environment, which don't support multicast by default. I've configured these hosts to use unicast Ganglia inter-cluster updates.

Gmetad uses unicast to speak to a single node in each cluster, so if you have a fairly straightforward network setup, the only firewall requirement is that the host running gmetad be able to reach port 8649 on any host running gmond. It's a symptom of an elegant design on the part of the Ganglia developers, I think, that there is so much automatic detection and so little overhead or impact inherent in this system. I can't remember the last time I picked up a new monitoring tool that installed and configured this easily.

Creating new graphs and adding new metrics are non-trivial undertakings, the price perhaps for getting so much functionality out of the box for so little work. To create new graphs from existing data you'll need to write a template in PHP, but the process is well documented [5] and should be easy enough for a decent sysadmin to follow. Adding new metrics can be accomplished in one of three ways. The first, and probably the easiest, is to use the included gmetric program, which will take input from any shell command and multicast it to all listing gmond nodes in the cluster in the same way gmond itself does. Any number of metrics can be added this way, via shell scripts running from cron, or hooks from the monitoring system of your choice.

The second and third options are to write a Ganglia plug-in in either Python or C. The process is pretty much the same for either type of plug-in, although the Python plug-ins require that gmond be compiled against the Python libs, and that requires you to have Python installed everywhere. For those of us who want to add new metrics while keeping the overhead as small as possible, there's the C option. But that's fodder for another article. Stay tuned.

Take it easy.

REFERENCES

[1] Cacti, "The Complete RRDTool-based Graphing Solution": http://www.cacti.net.

[2] Drraw: http://Web.taranis.org/drraw/.

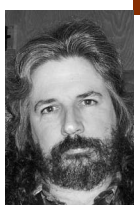
[3] Ganglia: http://ganglia.sourceforge.net/.

[4] Edward Tufte, a dude who knows a thing or two about presenting information: http://www.edwardtufte.com/tufte/.

[5] http://sourceforge.net/apps/trac/ganglia/wiki/Custom_graphs.

ROBERT G. FERRELL

/dev/random



Robert G. Ferrell is an information security geek biding his time until that genius grant finally comes through.

rgferrell@gmail.com

I SPENT, ALL TOLD, ABOUT NINE HOURS this weekend in the saddle of a zero-turn-radius beast (Ol' Yeller) doing some of the fall mowing of my property east of San Antonio. All that time dodging choking dust clouds, aggressive insects, and haphazardly flung vegetation gave me ample opportunity to contemplate the current state of information security—which is, as usual, dismal.

If this seems an unreasonably pessimistic appraisal, spend a few minutes perusing any periodical or Web site devoted to infosec and you must quickly come to the same conclusion. The reasons for this sad state of affairs are myriad: poor security engineering practices, failure on the part of corporations and government institutions to collect and analyze security-relevant operating system events, little or no network filtering/packet inspection, appalling Internet hygiene, and world-class gullibility being among the most prominent.

Besides the host of obvious issues, however, there is an insidious nest of more subtle creepy-crawlies squirming beneath the floorboards. One of these abominations recently raised its misshapen head and wriggled behind the wainscoting, there to bore yet more holes in the already sagging load-bearing timbers of our information economy. I refer—in case you're wondering where, if anywhere at all, I was going with this tortured metaphor—to security flaws masked as marketing strategies.

The latest half-baked idea from a popular microprocessor vendor, wherein they will sell you a processor that only functions as designed *after* you've purchased an additional code to unlock the crippled parts, is quite possibly the most offensive and ill-conceived travesty since the Digital Millennium Copyright Act in one fell swoop made *a priori* criminals of most of the entertainment-consuming population. Every time I think high-tech marketing can't possibly get any more inane it suddenly does, and without even breaking a sweat.

The problem with proprietary hardware and software is that you really don't know what you're purchasing. We as consumers long ago bought into the "we can't tell you how it works because that's a trade secret" corporate philosophy lock, stock, and barrel under the cover of free market capitalism; now we're stuck with the consequences of this appalling lack of collective judgment. The area where the black box principle takes one of its

heaviest tolls on society is security, or, rather, the profound absence thereof. Let's employ one of the most familiar and ironic of all Internet polemic tools to examine this: a reference book known as "The Dictionary," as though, *Highlander*-esque, there can be only one. Mine defines *security* this way: "The quality or state of being secure: as: (a) freedom from danger: safety; (b) freedom from fear or anxiety." It is the latter meaning I want to address here.

Fundamentally, anxiety is the absence of contentment. If you are satisfied with things the way they are, you aren't going to be anxious. Contentment itself can be brought about by following either of two protocols: (1) be certain that everything is going your way; (2) simply don't care whether it is or not.

Threading our way back up this chain of reasoning to the original stated premise, then, one path to security is apathy. That is the path the vast majority of computer users choose to take. Security means never having to say you bothered.

How does this relate to the vending of broken microprocessors? Hear me out. The manufacturer says the code they will gladly sell you to un-hobble the processor you've already paid for (apparently that initial outlay is merely a deposit) unlocks more level 2 cache and hyperthreading. Fine. That's just the part they're *telling* you about. What else is crippled, either intentionally or as a victim of collateral damage? You don't know? Neither do I. That uncertainty should trigger all sorts of alarms about potential security flaws but it won't, because *not caring is not worrying*. Security by apathy. Don't worry, be hapless.

Black boxes function or they don't, but this is not for you and me to question or influence. To paraphrase Arthur C. Clarke, any sufficiently obfuscated technology is indistinguishable from magic. Vendors are effectively telling us not to worry about how their products function: it's magic and you wouldn't understand, anyway. Trust us to have baked robust security right in. No, we won't reveal exactly what kind of security it is, but believe us, it's like really, really secure.

Let's take a little flight of fancy here (please limit your carry-on items to two per passenger) and imagine what might happen if this marketing philosophy were extended to other types of security-related consumer goods.

[*Wavy dissolve to smiling, attractive blonde holding brightly-colored product package*]

Congratulations on your purchase of a new EverJam 9000 Burglar-Proof Padlock. Enclosed in the packaging you will find a coupon for 10% off on the optional EverKey Pro Upgrade that allows you to open your new lock at any time, rather than once per day. Please do not read the warranty waiver, then sign, date, and return in the postage-paid envelope.

[*Quick wipe to photo montage of serene meadows full of flowers*]

Thank you for choosing the Cochlea Blaster Home Security System, now with extra-strength Decibels. Please call 1-888-URPUNKD to activate your account. Have your account identifier, birth certificate, three major credit cards, bank routing number, Social Security card, and blood type ready. At the prompt, press "1" for 24-Hour Monitoring (additional charge applies); press "2" if you want the optional "Manual Silencing" package (additional charge applies); press "3" if you want to disable "No Open" notifications (additional additional charge applies).

I could go on, but the equine is beginning to stiffen up.

[*Fade to black*]

book reviews

ELIZABETH ZWICKY, WITH RIK FARROW AND SAM STOVER

A few introductory notes. First, this isn't just the security issue; it's also the issue before Christmas, and somehow, despite the fact that I'm in the middle of a heat wave as I type, it's Christmas that caught me up as a theme. So I've thrown some less technical books into the mix.

Second, this issue marks my adoption of eBooks as a reviewing medium. A lot of factors drove me in that direction. There's the arrival of the iPad, which may be an overgrown phone, but it's a form factor that works for me for reading books. There's the sheer waste of shipping paper around, when I don't always like every book enough to review it. And then there's the fact that the first book I'm reviewing weighs more than my laptop, and at that is printed on paper thin enough to make turning pages challenging. So far, I'm enjoying the eBook experience, aside from distribution issues, which are often different for review copies than for release copies. I choose to always review final copy, but eBooks allow me to review that before the paper copies print, which usually means the eBook isn't officially available either.

UNIX AND LINUX SYSTEM ADMINISTRATION HANDBOOK, FOURTH EDITION

Evi Nemeth, Garth Snyder, Trent R. Hein, Ben Whaley, et al.

Prentice Hall, 2010. 1232 pp.
ISBN 978-0-13-148005-6

The fourth edition is also the twentieth-anniversary edition, which means that words like "classic" are pretty much inevitable. It also means that lots and lots of people already have opinions about previous editions. If you are such a person, you can stop reading now; this edition is unlikely to change your mind in any direction. Probably the only thing you need to know is that it covers old-style big-company UNIX (Solaris, HP-UX, AIX) and Linux, but not any BSD variants. I miss the BSD variants, particularly because Mac OS is a BSD variant; on the other hand, I really wouldn't want the book to be any longer than it already is.

If you haven't previously encountered the book, its strengths are in its technical coverage. It assumes that you know the absolute basics of how to use UNIX, and covers most of the bases you'll need to know to run a UNIX-based site, from scripting through mail sending and security. It's not evenly deep; it will teach you a great deal about mail, and hardly anything about cryptography. But it will teach you something about close enough to everything. There's a degree of old-fashioned UNIX attitude, which leavens the experience (and, to my taste, occasionally descends into annoying snark, but one person's well-founded negative opinion lightly expressed is the next person's annoying snark, and, in any case, it's lightly sprinkled).

Its weaknesses are, as always, the flip side of its strengths. It often fails to structure information, particularly non-technical information, so that you get a list of rules of thumb for something (security, or laying out file systems, or doing backups) but no overarching principles. The result is that you can use it to figure out how to do something, but, in most cases, you're not going to get much useful help on what you ought to be doing in the first place. (For that, see *The Practice of System and Network Administration*, ISBN 978-0321492661.)

Also, because it covers everything, it has trouble hitting a "just-right" level of abstraction. Some things are covered quickly enough to be indigestible for a beginner. A combination of factors, ranging from sheer lack of page space to a heritage dating back to days when all text all the time was the standard, means that it is critically short on illustrations.

As a resource, it's much more efficient than trying to paw through vendor documentation, and it covers more territory. Most system administrators will want a copy to look things up in, or learn about a new area.

COOKING FOR GEEKS

Jeff Potter

O'Reilly, 2010. 398 pp.
ISBN 978-0-596-80588-3

RATIO: THE SIMPLE CODES BEHIND THE CRAFT OF EVERYDAY COOKING

Michael Ruhlman

Simon and Schuster, 2009. 272 pp.
ISBN 978-1-4165-7172-8

In case you're Christmas shopping for a cook or somebody who might want to be a cook, here are two immensely geeky cookbooks. Or at least, one of them makes an explicit claim to be both intended for geeks and a cookbook; the other one wasn't even filed with the cookbooks at my bookstore and doesn't specify an audience, but don't let that fool you.

I started out feeling a bit hostile about *Cooking for Geeks*. Yeah, I call myself a geek, but it's not like it defines my life. I think I cook like a human being, and I am suspicious of things that treat "geek" like it's necessarily a meaningful group identifier. Plus, does the world really need more cookbooks? More introductory cookbooks, even?

It's not that my suspicions were totally unfounded, but I was won over pretty rapidly. Mostly, the book covers territory that other cookbooks don't, and regardless of how you feel about the title, it's a fun read for people at almost any level of interest in food. There are interviews with technical/foodie types you've heard of and you haven't, and it takes approaches you're not going to find anywhere else, such as discussions of protein denaturation and the biomechanics of taste.

Will it teach you to cook? Eh, maybe. Hard to tell. Most people I know either cook or they don't, and I'm not certain that a book is going to move people from one camp to another, but this book is going to intrigue the cooks. In addition, it certainly could have saved many semi-cooks I know from a number of unpleasant surprises.

Will it teach you new tricks? Oh, surely. Unless you already work for a cutting-edge restaurant, there are techniques here you haven't tried, probably haven't heard of, and almost certainly didn't think you could achieve at home. Some of them require more daring than others, some of them are actually actively recommended against (useful in itself), but you're certainly not going to walk away thinking, "I already knew all of that," and you probably will be thinking, "That sounds like fun, I might try that."

However, any geeky cook is also going to want *Ratio*. It's almost as technical as *Cooking for Geeks* and, in a way, is much more tightly focused. There's something a little contradictory about calling a book that ranges from bread to soup to custard "focused," but it's a much more integrated, less multi-threaded experience. It takes a wide range of recipes that can be thought of as ratios, provides ratios, explains them (both books talk about baking and gluten and air and the difference between popovers and crepes, which is a lot less than you'd think when you look at them, or eat them), and talks about how to modify them. It's a bolt of clarity, which will open whole new vistas of experimentation.

THE LEGO TECHNIC IDEA BOOK: SIMPLE MACHINES

Yoshihito Isogawa

No Starch Press, 2010. 157 pp.
ISBN 976-1-59327-277-7

This is an eccentric but seductive book. It consists almost entirely of pictures of Lego constructions, with the occasional number. Most of the Lego constructions are very simple machines: a few gears, or some pulleys, or some tracks. They build up to combinations that do things, examples of lots of things you could do with a single car chassis, or many, many different doors.

People with any reasonable amount of interest in Lego are rapidly sucked in; they start going "Oh! That's clever!" or "Hmm. What could I do with that?" The simplest mechanics are shown in pretty much every possible configuration, so if you don't have the precise parts shown in later constructions, you should be able to figure out how to replace them. Many of the pieces used are in non-classical Technic colors, which helps suggest improvisation. I was worried that you might need all sorts of Technic to have a good time with the book, but one good-sized Technic set produced enough parts to play with the book satisfyingly, if not to build every single thing.

It could actually use a few more words, or at least some words repeated; there are titles in the table of contents, but not on the pages. The titles help in figuring out what's being shown, and they provide

vocabulary for kids; mine was frustrated by seeing pictures of a group of things and not knowing what they were called. Still, she enjoyed it greatly. A child who cared more about understanding the author's intent would have a harder time.

CONFIGURATION MANAGEMENT BEST PRACTICES

Bob Aiello and Leslie Sachs

Addison Wesley, 2010. 217 pp.
ISBN 978-0-321-68586-5

"Configuration management" here means primarily software configuration management, not system configuration management, in case you're a hopeful system administrator. The wars it carefully stays out of are Ant vs. Maven, not cfengine vs. Puppet.

If you're interested in configuration management as a career, this will give you a good framework in which to think about how configuration management goes together and how it fits into companies. It also has a nice variety of anecdotes, and a sensible approach to psychology as a part of configuration management.

If you're not already passionate, it's not likely to convince you, unfortunately. It's unevenly edited, gripping in spots and totally flat in others. And I know I just complained about *UNIX and Linux System Administration* being short on illustrations, but somebody seems to have decided that *Configuration Management* needed illustrations for their own sake. The resulting picture of the globe did not actually deepen my understanding of distributed configuration management issues.

YOUR MONEY: THE MISSING MANUAL

J.D. Roth

O'Reilly Media, 2010. 324 pp.
ISBN 978-0-59-680940-9

REVIEWED BY RIK FARROW

As sometimes happens, this book appeared in the post, out of the blue. And while I felt comfortable about my own finances, I thought that I knew of some people who might profit by reading it. The book is better than I had expected, and I wound up learning a lot by reading it.

Roth started out, as he writes, living paycheck-to-paycheck and getting deeper in debt. He began researching to learn how to extricate himself from his predicament, then started a Web site that shares his findings: GetRichSlowly.org.

Understanding the math is simple, Roth writes, but controlling your emotions and habits is hard. Roth is always gentle, and very clear, as he lays out techniques for reducing debt, budgeting, saving, investing, and even retiring. I picked up his book wondering whether someone buried in credit card and student loan debt would appreciate getting it as a gift, and the answer is absolutely. For myself, I learned more about investing, and will reread his section on buying a car the next time that comes up.

INSIDE CYBER WARFARE

Jeffrey Carr

O'Reilly Media, 2009. 220 pp.
ISBN 978-0-596-80215-8

REVIEWED BY SAM STOVER

The author of this book acknowledges his role as the "Principal Investigator of the open source intelligence effort Project Grey Goose." I'm going to ignore the possible connotations of this fact, but anyone with a predisposition regarding Project Grey Goose is likely to apply it to this book, especially as the project is referred to often.

Cyber warfare is certainly a high-impact buzzword in today's world, and this book does a reasonable job of discussing the topic in a way that is relevant to techies and managers alike. Chapter 1 starts out by "Assessing the Problem," which is no easy task, since everyone seems to have their own idea of what qualifies as cyber warfare. One important point, that I happen to agree with, is that too many people want to separate cyber crime from cyber warfare, and this book posits that too many bad guys dabble in both.

Chapter 2, "Rise of the Non-State Hacker," discusses the 2008 Russian-Georgian conflict, along with Israeli-Arabic cyber attacks. Specific events are cited to show different methods used by attackers to disrupt, deface, redirect, or otherwise negatively impact their adversary. The chapter ends with the cold hard truth that in China and Russia, attacks against other countries are just not prosecuted by law enforcement (LE). In my experience, this rings true—if "local" citizens are not being attacked, good luck getting international LE support.

Chapter 3 focuses on how different countries deal with cyber attacks. A number of attacks are discussed, as well as any legal action brought to bear. The chapter rounds out with eight mini-scenarios that have occurred in eight different countries since the Russia-Georgia conflict. This

is followed by a return to the question, what exactly is considered a cyber attack? Chapter 4 also deals with legal issues, but is actually a compressed thesis by another author (Lt. Cdr. Matt Sklerov from the DoD). I think this chapter was my favorite, as it discussed when "states may lawfully respond to cyber-attacks in self-defense." "Hacking back" has been a hotly debated topic for, well, forever, and I really found the legal issues explained in this chapter to be interesting, although I'm not sure I completely agree with everything that's said here.

Chapter 5 uses several different incidents, including the Korean DDoS attacks, the 2009 Ingushetia conflict and the Russia-Georgia conflict, to show the value in intelligence gathering when trying to piece together the "big picture." Chapter 6 shows how social networking services play a relevant part in both attack collaboration and attack surface. Chapter 7 deals mainly with how the bad guys are able to work within the confines of the Internet. Bulletproof Hosting is dissected and explained in the context of the Russian-Georgian conflict.

Chapter 8 dives into "Organized Crime in Cyberspace," which is another hotly debated topic, and it's not really surprising that the Russian Business Network (RBN) is the example used for discussion. There is a lot of speculation on what the RBN was/is, where it came from, where it went, and who was involved.

Attribution of attacks and attackers is a huge part of cyber attack investigations, and Chapter 9 goes through a number of methods and sources of information. Open source data such as AS and BGP info, as well as WHOIS, and darknet monitoring are presented. For the techies in the group, this is a decent, albeit short, chapter with a little technical meat to it. Chapter 10, "Weaponizing Malware," also has some good tidbits such as SQL injections, iframe attacks, rootkits, and social engineering attacks. Chapters 11–13 deal with military doctrine, early warning models, and policy advice, respectively. Not to say that I found these chapters boring, but, well, I kinda did, maybe because the preceding two had some decent technical stuff.

The book has lots of quotes and references, which I found a little tedious, but is chock full of details and supporting material, which lots of other books tend to lack. I read this book in softcopy (O'Reilly's Safari), so I don't have a feel for how hefty the book was, but most chapters seemed very short. Lots of references to Project Grey Goose—it seems that a lot of the data/findings were contributed by that effort. There are definitely places where the author's opinion is stated as if it were fact, but I guess that's par for the course. Overall, this is a solid, but brief, look at cyber warfare, with a heavy emphasis on the Russian-Georgian conflict.

USENIX notes

USENIX MEMBER BENEFITS

Members of the USENIX Association 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 such topics as security, Perl, networks, and operating systems, book reviews, and summaries of sessions at USENIX conferences.

ACCESS TO ;LOGIN: online from October 1997 to this month: www.usenix.org/publications/login/

ACCESS TO VIDEOS from USENIX events in the first six months after the event: www.usenix.org/publications/multimedia/

DISCOUNTS on registration fees for all USENIX conferences.

SPECIAL DISCOUNTS on a variety of products, books, software, and periodicals: www.usenix.org/membership/specialdisc.html

THE RIGHT TO VOTE on matters affecting the Association, its bylaws, and election of its directors and officers.

FOR MORE INFORMATION regarding membership or benefits, please see www.usenix.org/membership/ or contact office@usenix.org; 510-528-8649.

USENIX BOARD OF DIRECTORS

Communicate directly with the USENIX Board of Directors by writing to board@usenix.org.

THANKS TO OUR VOLUNTEERS

Ellie Young, Executive Director

As many of our members know, USENIX's success is attributable to a large number of volunteers, who lend their expertise and support for our conferences, publications, good works, and member services. They work closely with our staff in bringing you the best there is in the fields of systems research and system administration. Many of you have participated on program committees, steering committees, and subcommittees, as well as contributing to this magazine. We are most grateful to you all. I would like to make special mention of the following individuals who made significant contributions in 2010.

Program Chairs

Randal Burns and Kimberly Keeton: 8th USENIX Conference on File and Storage Technologies (FAST '10)

Ethan L. Miller and Erez Zadok: First USENIX Workshop on Sustainable Information Technology (SustainIT '10)

Margo Seltzer and Wang-Chiew Tan: 2nd Workshop on the Theory and Practice of Provenance (TaPP '10)

Miguel Castro and Alex C. Snoeren: 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI '10)

Michael J. Freedman and Arvind Krishnamurthy: 9th International Workshop on Peer-to-Peer Systems (IPTPS '10)

Aditya Akella, Nick Feamster, and Sanjay Rao: 2010 Internet Network Management Workshop/Workshop on Research on Enterprise Networking (INM/WREN '10)

Michael Bailey: 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More (LEET '10)

Geoff Lowney and David Patterson: 2nd USENIX Workshop on Hot Topics in Parallelism (HotPar '10)

Paul Barham and Timothy Roscoe: 2010 USENIX Annual Technical Conference (USENIX ATC '10)

John Ousterhout: USENIX Conference on Web Application Development (WebApps '10)

Jiri Schindler: 2nd Workshop on Hot Topics in Storage and File Systems (HotStorage '10)

Erich Nahum and Dongyan Xu: 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '10)

Bruce Maggs, Andrew Tomkins, and Balachander Krishnamuthy: 3rd Workshop on Online Social Networks (WOSN 2010)

Ron Minnich and Eric Van Hensbergen: FAST-OS Workshop

Dave Cohen and Jason Stowe, 1st USENIX Cloud Virtualization Summit

Alva L. Couch: Configuration Management Summit

Ian Goldberg: 19th USENIX Security Symposium (Security '10)

Jelena Mirkovic and Angelos Stavrou: 3rd Workshop on Cyber Security Experimentation and Test (CSET '10)

Charlie Miller and Hovav Shacham: 4th USENIX Workshop on Offensive Technologies (WOOT '10)

Doug Jones, Jean-Jacques Quisquater, and Eric Rescorla: 2010 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE '10)

Nadav Aharony, Yaniv Altshuler, and Yuval Elovici: 2010 Workshop on Collaborative Methods for Security and Privacy (CollSec '10)

Kevin Fu, Tadayoshi Kohno, and Avi Rubin: 1st USENIX Workshop on Health Security and Privacy (HealthSec '10)

Wietse Venema: 5th USENIX Workshop on Hot Topics in Security (HotSec '10)

Andrew Jaquith and Khalid Kark: Fifth Workshop on Security Metrics (MetriCon 5.0)

Remzi Arpaci-Dusseau and Brad Chen: 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI '10)

Yvonne Coady and James Mickens: Workshop on Supporting Diversity in Systems Research (Diversity '10)

Greg Bronevetsky, Kathryn Mohror, and Alice Zheng: Workshop on Managing Systems via Log Analysis and Machine Learning Techniques (SLAML '10)

Paulo Verissimo and Hakim Weatherspoon: Sixth Workshop on Hot Topics in System Dependability (HotDep '10)

Frank Bellosa and Trishul Chilimbi: 2010 Workshop on Power Aware Computing and Systems (HotPower '10)

Mike Dahlin and Milan Vojnovic: 2010 Workshop on the Economics of Networks, Systems, and Computation (NetEcon '10)

Ralf Huuck, Gerwin Klein, and Bastian Schlich: 5th International Workshop on Systems Software Verification (SSV '10)

Rudi van Drunen: 24th Large Installation System Administration Conference (LISA '10)

Invited Talks/Special Track Chairs

Hakim Weatherspoon: Work-in-Progress Reports (WiPs) and Posters at FAST

David Pease: Tutorial Chair at FAST

Charles Killian: Posters at NSDI

Shan Lu: WiPs and Posters at USENIX Annual Tech

Dan Boneh, Sandy Clark, and Dan Geer: Invited Talks at USENIX Security

Patrick Traynor: Posters at USENIX Security

Carrie Gates: Rump Session at USENIX Security

Jon Howell: Posters at OSDI

Sam King, Shan Lu, and Emmett Witchel: Research Vision Session at OSDI

Æleen Frisch, Doug Hughes, and Amy Rich: Invited Talks at LISA

Kent Skaar: Workshops at LISA

Chris St. Pierre: Guru Is In sessions at LISA

Rudi van Drunen: WiPS and Posters at LISA

Other Major Contributors

John Arrasjid, David Blank-Edelman, Matt Blaze, Gerald Carter, Clem Cole, Alva Couch, Rémy Evard, Brian Noble, Niels Provos, and Margo Seltzer for their service on the USENIX Board of Directors

Jeffrey Bates, Steven Bourne, Clem Cole, Timothy Lord, Jim McGinness, and Keith Packard for serving on the USENIX Awards Committee

Rob Kolstad, Don Piele, Brian Dean, and Percy Liange for their work with the USA Computing Olympiad, co-sponsored by USENIX

Dan Geer and Theodore Ts'o for serving on the Audit Committee

Rémy Evard for chairing the USENIX Board Nominating Committee

Eddie Kohler for his HotCRP submissions and reviewing system

Jacob Farmer of Cambridge Computer Services for his sponsorship of the USENIX Education on the Road series and for organizing the Storage Pavilion and Data Storage Day at LISA

John Arrasjid, Mark Burgess, Tina Darmohray, Duncan Epping, Steve Kaplan, and Carolyn Rowland for writing the three Short Topics books published by USENIX in 2010

Matthew Sacks and Matt Simmons for blogging about USENIX activities

Statement of Ownership, Management, and Circulation, 9/30/10

Title: ;login: Pub. No. 0008-334. Frequency: Bimonthly. Subscription price $125.
Office of publication: USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.
Headquarters of General Business Office of Publisher: Same. Publisher: Same.
Editor: Rik Farrow; Managing Editor: Jane-Ellen Long, located at office of publication.
Owner: USENIX Association. Mailing address: As above.
Known bondholders, mortgagees, and other security holders owning or holding 1 percent or more of total amount of bonds, mortgages, or other securities: None.
The purpose, function, and nonprofit status of this organization and the exempt status for federal income tax purposes have not changed during the preceding 12 months.

Extent and nature of circulation	*Average no. copies each issue during preceding 12 months*	*No. copies of single issue (Oct. 2010) published nearest to filing date of 9/30/10*
A. Total number of copies	4981	5253
B. Paid circulation		
Outside-county mail subscriptions	2961	3007
In-county subscriptions	0	0
Other non-USPS parcel distribution	1555	1532
Other classes	0	0
C. Total paid distribution	4516	4539
D. Free distribution by mail		
Outside-county	0	0
In-county	0	0
Other classes mailed through the USPS	74	69
E. Free distribution outside the mail	352	569
F. Total free distribution	426	638
G. Total distribution	4962	5177
H. Copies not distributed	39	76
I. Total	4981	5253
Percent Paid and/or Requested Circulation	92%	89%

I certify that the statements made by me above are correct and complete.
Jane-Ellen Long, Managing Editor

conference reports

THANKS TO OUR SUMMARIZERS

19th USENIX Security Symposium (USENIX Security '10)

Washington, DC
August 11–13, 2010

OPENING REMARKS AND AWARDS PRESENTATION

Program Chair: Ian Goldberg, University of Waterloo

Summarized by Rik Farrow (rik@usenix.org)

Ian Goldberg thanked the USENIX staff and the program committee, then said there were a record number of submissions for this year's conference: 207 papers were submitted. Five were rejected for double submissions or plagiarism, and 42 more papers were rejected in the first round of reviews. Each Program Committee member read 20–22 papers, with David Wagner reading 38. In the end, 30 papers were accepted, with Capsicum (Watson et al.) winning Best Student Paper, and Vex (Bandhakavi et al.) Best Paper.

KEYNOTE ADDRESS

- ***Proving Voltaire Right: Security Blunders Dumber Than Dog Snot***

Roger G. Johnston, Vulnerability Assessment Team, Argonne National Laboratory

Summarized by Veronika Strnadova (vstrnado@unm.edu)

Johnston began by pointing out numerous, all too common mistakes that his team encounters when assessing security vulnerabilities, along with some countermeasures. Many of these mistakes are avoidable and many vulnerabilities are fixable, but Johnston said that the big problem comes from the fact that a lot of people don't exploit security resources.

One ineffective response to a security threat is the use of what Johnston dubbed "security theater," the practices which involve providing a "feel-good effect" for the public instead of making a true effort to increase security. A glaring example of this is the no-fly list or color-coded security threat level we see at airports. While security theater may provide comfort to some people, many vulnerabilities are being overlooked—Johnston and his team were able to tamper with voting machines, spoof GPS devices, break into containers with expensive cargo (breaking and then replacing seals), proving that common sense is still a much-needed tool not being used in security.

Johnston talked about some alarming vulnerabilities that arise from oversights such as not doing background checks on IAEA nuclear site inspectors, not setting a microprocessor's security bit, or not masking passwords and sensitive data. Most importantly, many people and/or companies don't take Johnston's advice or implement even the simplest security measures. Johnston said that

this often comes from a fear that admitting vulnerabilities means admitting weakness or ineffectiveness. In response to a question from Rik Farrow, he said that when people don't take his advice, he doesn't take it personally and that his team offers solutions but can't expect everyone to implement them.

Peter Neumann asked about implementing common sense, and Johnston suggested that people in physical and cyber security can help and learn from each other to fix security blunders. The two cultures need to learn to work together to avoid making the same mistakes over and over. He said his team has learned that telling people that not everything can be fixed helps alleviate the pressure of improving security. Someone else wondered whether people get fired for vulnerabilities his team finds. Johnston said this only happens after a discovered vulnerability gets exploited. According to Johnston, this is the worst time to fire people (usually not the ones at fault) who can help counter the security threat, but it happens often in physical security. Matt Blaze was impressed with the work done to switch votes in a voting machine, but wondered if it would have been easier to replace the printed overlay. Johnston agreed this was a possibility, but pointed out that his hardware attack could be done at more points before voting.

Johnston said that security needs to be thought about when designing a product, not as an afterthought. Security is not usually something that can be added on, and adding on "layers" of security only makes it more difficult to monitor when security has been broken. Countermeasures are often simple—seals should be unique and difficult to break, the order of candidates in voting machines should be randomized, and people within companies can often find vulnerabilities in security at no extra cost to the company.

No matter how many suggestions Johnston's team makes, the major problem is that there is often no interest in implementing them. There is a need for research-based countermeasures to security threats, but "security theater" is easier. The laziness and blind faith in security "authorities" who often have no experience in dealing with security are the first vulnerabilities that need to be changed.

PROTECTION MECHANISMS

Summarized by Zhiqiang Lin (zlin@cs.purdue.edu)

- ***Adapting Software Fault Isolation to Contemporary CPU Architectures***
  David Sehr, Robert Muth, Cliff Biffle, Victor Khimenko, Egor Pasko, Karl Schimpf, Bennet Yee, and Brad Chen, Google, Inc.

Software Fault Isolation (SFI) is an effective approach to sandboxing untrusted binary code. Native Client (NaCl) is an interesting use case of SFI for running native code in Web applications. Through extending NaCl, David Sehr presented new schemes on how to make SFI effective on some contemporary CPU architectures, namely ARM and x86-64. Their implementations on these two architectures are called ARM-SFI and x86-64-SFI, which are the best known SFI implementations with significantly lower overhead (under 5% on ARM and 7% on x86-64) than previous systems.

Their work was prompted by Web application scenarios in which programmers tend to use their own favorite language but like to have some features such as screening malicious instructions, system calls moderated by a virtualized OS, and performance within 5% of native code execution. With NaCl, users do have these benefits to run applications such as Star Wars and Nexuiz (an openGl Quake) inside the Web browser. Sehr provided some background on SFI such as creating an untrusted memory segment and using instructions to enforce segment boundaries, the control safety and data safety in SFI, and the SFI implementation.

In particular, for SFI implementation, Sehr outlined two basic approaches: using hardware support (e.g., x86 segmentation), and inline guard instruction sequences which require aligned instruction blocks. He also talked about some background on stack pointer optimization proposed by McCamant and Morrisett. After covering the architectures of x86-32, x86-64, and ARM at a high level, he described how they implemented their ARM-SFI and x86-64-SFI. More specifically, he talked about how they ensure the control safety, data safety, and stack updates for untrusted code in ARM and x86-64, respectively. For the performance, they are very pleased with the ARM-SFI, and the results are fairly consistently around an average of 5%. However, interestingly, the performance of x86-64 is bimodal: where code size is important, overhead rises to 30%; where code size is not significant, overhead is low. Their code is available at http://code.google.com/p/nativeclient.

Peter Neumann commented that there is another piece of work called BackerSField by Joshua Kroll. Neumann wondered whether Sehr has discussed this with Kroll, as Google had hired him this summer. Sehr responded that they did have brief discussions on sandboxing improvement (e.g., performance). Another person asked about the wisdom of research on solving problems we probably already have solutions for. Sehr answered that they first looked at the problem from a performance perspective, and they found there are still gaps to be closed. Also, there are still some theoretical problems to explore, such as the formal models of instruction sequences in different architectures they are targeting, and how to make the validator do more formal verifications.

- ***Making Linux Protection Mechanisms Egalitarian with UserFS***
  Taesoo Kim and Nickolai Zeldovich, MIT CSAIL

UserFS "provides egalitarian OS protection mechanisms in Linux [and] allows any user to allocate . . . UNIX user IDs, to use chroot, and to set up firewall rules in order to con-

fine untrusted code." Taesoo Kim began his talk on how to build secure applications, which is simple in principle, but it is difficult in practice (unless you're a root user) because normal users cannot create new principals and cannot reduce privileges. Thus, his talk is about how to help programmers reduce privileges and enforce security policy in Linux by allocating and managing UIDs.

Kim used DokuWiki as a running example to illustrate their techniques. DokuWiki is a PHP-based wiki, and it runs as a single UID but has a number of users. As such, DokuWiki has to perform ACL checks when different users access particular files. If a programmer missed any ACL checks, it could lead to an insufficient permission check vulnerability. The goal of UserFS is to allow any application to use existing protection mechanisms without root privileges, such as creating a new principal, reusing existing protection mechanisms, and using chroot and firewall mechanisms. One key idea in UserFS is to represent user IDs as files in a /proc-like file system, thus allowing applications to manage user IDs like any other files, by setting permissions and passing file descriptors over UNIX domain sockets. There are several challenges in making user IDs egalitarian, including how to reuse UIDs, how to make UIDs persistent, accountability, and resource allocation.

The authors have implemented UserFS as a single kernel module with 3000 lines of code on Linux 2.6.31 using Linux Security Module, Netfilter, and the Virtual File System. In their evaluation, they have modified five applications to take advantage of UserFS. By changing just tens to hundreds of lines of code, they prevented attackers from exploiting application-level vulnerabilities, such as code injection or missing ACL checks in a PHP-based wiki application. Also, UserFS incurs no performance overhead for most operations, making it practical to deploy on real systems. Kim also discussed the limitations of UserFS, such as UID generation numbers only tracked for setuid binaries, and GID allocation not implemented in their current prototype; their future work is to allow a process to have multiple concurrent UIDs.

Someone asked why UserFS didn't store IDs for other files instead of only tracking the generation IDs for setuid binaries. Kim answered, "Because of performance." Someone asked about the impact on the resource limit for UserFS if the resource quota turns on. Kim replied that the programmer in that case has to use set on the resource system call in Linux. The third question concerned comparisons with Plan-9 from Bell Labs—more particularly, on the fact token with the notion of user ID as files, and on who guards the permissions in UserFS. David Reed said that UserFS is a nice mechanism and was curious about the generality of UserFS when compared with state-of-the-art capabilities, cross-domains, etc.

- ***Capsicum: Practical Capabilities for UNIX***
  Robert N.M. Watson and Jonathan Anderson, University of Cambridge; Ben Laurie and Kris Kennaway, Google UK Ltd.

  Awarded Best Student Paper!

Robert N.M. Watson presented Capsicum, a lightweight OS capability and sandbox framework, which extends the POSIX API and provides several new kernel primitives (e.g., sandboxed capability mode and capabilities) and a user-space sandbox API to support object-capability security for UNIX-like OSes. It supplements rather than replaces DAC and MAC.

Watson first described the paradigm shift from multi-user machines to multi-machine users and compartmentalized applications, and from DAC/MAC-centric access control to sandboxing. We are living in a world of browsers which can visit many Web sites (e.g., Webmail, YouTube, bank account) with very different technologies (e.g., traditionally static Web page, dynamic Web pages, virtual machines, and scripting languages). But Web browsers do have security vulnerabilities in large quantities. Watson mentioned the existing work, including microkernels, MAC, and Type Enforcement, to motivate their capability system. A capability is an unforgeable token of authority, and it supports delegation-centric access control. Capsicum supports capabilities with refined file descriptors with fine-grained rights, has a capability mode in which the sandbox denies access to the global namespace, and contains libcapsicum, a library to create and use capabilities and sandboxed components. To demonstrate Capsicum, the authors have added self-compartmentalization to a number of UNIX applications and core system libraries, including tcpdump, dhclient, and gzip using Capsicum. In collaboration with Google, they also have adapted the Chromium Web browser to use Capsicum, showing significant programmability and security benefits over its existing use of UNIX DAC and MAC security primitives. They prototyped Capsicum on FreeBSD 8.x, and their experimental code is BSD-licensed.

Peter Neumann asked about the dichotomy between capabilities and mandatory access control, given that, historically, systems in the 1970s that adopted this approach worked. Watson answered that the two composed quite well, but that things might prove more interesting in the case of applications already constrained by Type Enforcement when the use of capabilities was indicated. Helen Wang commented that capability-based sandboxing is definitely the right way to go, especially for the Web browser. She observed the similarity between the Chromium browser structure presented and the Gazelle project, as well as the observation regarding multi-user vs. single-user OSes. Watson responded that the browser architecture used under Capsicum is the model already present in Chromium, but that Chromium would benefit from much more use of sandboxing; another concern is how to address the windowing system. One exciting change has been in the use of new security models in mobile phones (such as the iPhone and Android),

where breaking existing applications was acceptable. Wang pointed out that the Web single-origin policy is one of the areas where Web browsers have gotten things right. Jinpeng Wei asked how to handle revocation capability in Capsicum. Watson answered that the revocation of capabilities is usually done through interposition, which is supported for userland capabilities (IPC objects) but not yet for kernel objects. Crispin Cowan asked for a comparison of AppArmor with Capsicum. Watson replied that AppArmor and Mac OS X's Seatbelt are very similar systems in terms of how policies are bound to applications, so a similar analysis would likely apply, but that a capability-oriented architecture had significant benefits.

INVITED TALK

- ***Toward an Open and Secure Platform for Using the Web***
  Will Drewry, Software Security Engineer, Google

  Summarized by Joshua Schiffman (jschiffm@cse.psu.edu)

Will Drewry began by discussing the design goals of Chrome OS and how they address the concerns of the typical user, who is unaware of security and unsafe browsing practices. In particular, he outlined three main areas: survivability of the system, data protection, and the openness of the platform. Starting from a baseline of a simple Linux distribution using Gentoo Linux's portage and no user-installed local applications, users only interact through the Chrome running on Xorg, which is supported by a mix of new and existing daemons underneath.

In terms of survivability, Google wanted a system that can recover from most forms of compromise, such as rootkits, trojans, BIOS modification, etc. To do this, they use a combination of mitigation techniques popular on clients such as ASLR, default non-execute heap and stack, sandboxing Chrome's renderers, and DAC. They also included protection features used on servers, such as a read-only root file system, restricted mount flags for non-rootfs, a set of kernel patches, and capabilities. They also use grsecurity and Tomoyo for mandatory access control instead of AppArmor or SELinux, because the Google team felt it was easier to keep the MAC policies in sync with feature development using them. Another tool they added was Breakpad, which is Google's crash dump logger that was linked into every binary.

Drewry then discussed how Chrome OS performs auto-updates by dividing the rootfs into an active and passive partition, which is then swapped after an update is applied to the passive partition and verified at boot. Updates are streamed to the disk using delta differences based on a block dependency graph, which Drewry noted was more efficient than something like bsdiff that requires the entire file system to be loaded into memory. Drewry also mentioned that Trusted Platform Module (TPM) was used only for its lockable NVRAM to provide rollback protection, but not for measuring files, since the Google team did not want to deal with managing the administrative passwords the TPM requires for those features.

Drewry then moved on to how the active partition was verified at boot time in order to assure users that Chrome OS is currently running. This approach uses a Static Root of Trust model to help prevent persistent basic attacks. The root of trust is a key that lives in read-only firmware, which verifies a subkey in a writeable firmware portion. This subkey is then used to verify the RW firmware, and this process is then repeated for the OS kernel and command line in the rootfs. Verification of the rootfs is done using a hash tree approach whereby each 4KB block is hashed and then 4KB of hashes is hashed repeatedly to form a single root hash that is then passed as a kernel line parameter. In this way, the OS can check the rootfs incrementally instead of all at once, which takes longer and slows the boot process.

Drewry then moved to the second design goal, which is protecting user data. Currently, users log into Chrome OS using their Google accounts or Google Account for your Domain. In the future, they would like to support OpenID providers, but mentioned that there are issues with passing attributes and that generic programmatic Web login is an open challenge. He also mentioned that users could browse without signing in and that such sessions are stored in a tmpfs. Actual user accounts have their data protected by a daemon known as Cryptohome, which manages user partitions encrypted with eCryptfs. This daemon is used in place of the standard eCryptfs utilities and handles offline authentication and partition keys. A user's passphrase is needed for decryption, but they mitigate brute force attempts by wrapping the derived key with a TPM key, which forces a brute force attacker to be subjected to slow hardware. If a TPM is not available, Colin Percival's scrypt for memory-hardening is used.

On the topic of openness, Drewry mentioned that Chrome OS is based on the open source Chromium OS and that the team frequently contributes back to the project. On the hardware side, a developer mode switch is specified to be under the battery to let knowledgeable users disable the boot process and load self-signed OS images. This process clears the TPM and zeroes RAM to prevent this from being abused by attackers that use boot shims to read memory, but does not prevent more sophisticated cold boot attacks. In the future, he said, they would like to integrate a platform-supported trusted UI and a peripheral firmware validation mechanism.

Someone asked about the time frame for official Chrome OS laptops, and Drewry answeredthat it would be sometime this year. Another audience member asked whether Chrome OS could function as a VM. Chrome OS would not, but Chromium is QEMU-friendly and has been shown to run in KVM and Virtual Box. One reason for not supporting VMs in Chrome OS is the difficulty in verifying the system, but one option would be to use some features in the EFI standard. Drewry also demoed a reboot of a Chrome OS laptop,

which took about 12 seconds due to the unmodified Dell firmware.

PRIVACY

Summarized by Tamara Denning (tdenning@cs.washington.edu)

- ***Structuring Protocol Implementations to Protect Sensitive Data***
  Petr Marchenko and Brad Karp, University College London

Petr Marchenko presented this work via Skype. The goal of this work is to help protect sensitive data in network applications such as Web servers. While SSL connections protect the confidentiality and integrity of customer data while in transit, they do not guarantee these properties if an attacker exploits a vulnerability in the Web application itself.

While breaking down the application into compartments and applying the principle of least privilege helps with security, current applications do not treat the session key as privileged information. Additionally, if an attacker compromises an unprivileged compartment, he can get a privileged compartment to sign arbitrary data in what is known as an oracle attack.

The researchers identified these attacks and developed some defenses. They employ a "session key barrier" by creating new unprivileged compartments after the session key negotiation. In addition, they created nine principles to prevent against oracle attacks. The researchers created a hardened version of the OpenSSH client and server and a drop-in replacement for the OpenSSL library that reduce the TCB and protect against the session key attack and all known oracle attacks.

Questions from the audience addressed the aspects of a Web application that are not protected by these defenses. For example, if user data is handled by an unprivileged compartment that can be compromised by an attacker, there is still a breach in the confidentiality of the user's data. The speaker commented that the work is focused on one specific aspect of the security of network applications and that other vulnerabilities may still exist.

- ***PrETP: Privacy-Preserving Electronic Toll Pricing***
  Josep Balasch, Alfredo Rial, Carmela Troncoso, Bart Preneel, and Ingrid Verbauwhede, IBBT-K.U. Leuven, ESAT/COSIC; Christophe Geuens, K.U. Leuven, ICRI

Josep Balasch presented this work on privacy for electronic toll systems. The EU has chosen to employ satellite-based electronic toll systems that consist of a GPS and GSM in an on-board unit (OBU) in a user's car. The system tracks the user's location and periodically sends information to the toll server. The goal of this work is to help design an electronic toll pricing system that respects user privacy, provides user verifiability of costs, and allows the providers to detect misuse of the system.

To protect a user's privacy, one solution is to have fee calculations be done on the OBU. However, this provides the user with the opportunity to tamper with price tables, subfees, and fee totals before their transmission to the provider. These attacks are in addition to the possibility of performing GPS spoofing or turning off the OBU. The researchers propose employing TPMs and homomorphic commitments to prices as part of a proof of adherence for the provider. These cryptographic techniques would be used in addition to spot checks performed by the provider (for example, via license plate cameras).

The researchers built a proof-of-concept OBU and demonstrated that the OBU and server time required are practical. For example, using 1-mile road segments and processing GPS strings every second, 1536-bit keys support a maximum vehicle speed of 124 mph. The researchers also showed that the amount of server processing required supports a reasonable number of vehicles on the road.

An audience question brought up a discussion of interoperability between EU countries and the different levels of interest in privacy in those countries. While the EU plan for an electronic toll system has no privacy requirements, some countries have expressed an interest in making the system privacy-respecting. Another question addressed the possibility of optimizations in the system implementation, such as using elliptic curves. Balasch said that while their system employed optimized assembly routines, no attempts were made to optimize the cryptography involved.

- ***An Analysis of Private Browsing Modes in Modern Browsers***
  Gaurav Aggarwal and Elie Burzstein, Stanford University; Collin Jackson, CMU; Dan Boneh, Stanford University

Elie Burzstein presented this work on a general survey of the characteristics of private browsing modes in different modern browsers and the characteristics of users who employ private browsing modes. The researchers gathered data on the browsing histories of browsers in private mode by purchasing ads and detecting whether or not a link displays as having been visited.

The researchers found that private browsing mode is most common in users of Safari and Firefox, and that private browsing is most often employed when visiting sites with adult content. The authors presented a theory that private browsing is so prevalent in Safari because the private mode indicator is discreet, and therefore easy to leave on accidentally.

The researchers also analyzed the private browsing mode behavior of common browsers by leveraging unit tests. Browser violations of indistinguishability after a private browsing session included SSL certificates and site-specific preferences. Additionally, popular browser extensions frequently do not check for private mode or alter their behavior accordingly. The authors propose manual review of extensions to check that they are privacy-respecting and an

opt-in model for extensions when running in private browsing mode.

Someone brought up the question of whether, since studies show that the primary use of private browsing mode is for adult content, researchers are not acknowledging or addressing the main uses of privacy technologies in their discussions and research. Another question regarded the threat model of this work, which assumes that the computer is under user control until the private browsing session begins, and therefore excludes scenarios such as IT-managed computers in work settings.

INVITED TALK

- ***Windows 7 Security from a UNIX Perspective***
  Crispin Cowan, Senior Program Manager, Window Core Security, Microsoft, Inc.

  Summarized by Adam J. Aviv (aviv@cis.upenn.edu)

"Windows" and "security" are not words normally placed in the same context, especially not at USENIX Security, but it was the primary focus of a very well-attended invited talk in front of a raucous crowd. In the introduction, Crispin Cowan was described as one of the most outspoken critics of Windows just five years ago, but as of 2008, he is on the "other side." As a project manager at Microsoft, he was brought in to help with security on an OS that needed it, and, in his own words, "It hasn't been disputed that they needed it."

Cowan's thesis is simple. Yes, back in the day, Windows had lousy security (if any), but now Windows is leading the way. He provided a few key examples of this; some received jeers from the crowd, others nods of approval. Overall, it was an exciting talk that held the attention of everyone in the room.

Cowan began by describing the state of the world prior to Microsoft's security revelation. From Windows 3.1, 95, 98, up to XP, "all code that got to run on the box had complete ownership of the box." There were no privileged or unprivileged users; everything essentially ran as root: "Run as non-root, good. Run as root, not so good." He noted that NT was fundamentally a secure OS, but Windows applications grew up in a world without privileges, and so the default user had to be administrator, which just defeated the whole purpose.

This issue was just a small part of the "coin-operated" computer design of the time, pushing functionality over security. This was fine until the rise of the Internet. Then it became about not just what a user can do, but what others could do. In 2002, a memo by Bill Gates was circulated, saying as paraphrased by Cowan, "You will learn security."

This brought about a number of changes in the Microsoft development mantra, namely the SDL (Secure Development Lifecycle). Cowan described the SDL revelation this way: "All that stuff they teach you in college . . . what if we did that? Turns out it works!" As an example, in 2003 Microsoft SQL Server had a serious buffer overflow which resulted in a "flash worm." In 2004, after the SDL, and an update of SQL Server, there was just one vulnerability in three years. He compared this to MySQL, which had twelve serious vulnerabilities in the same time period.

Another example of Windows leading the way was the recompiling of Windows XP SP2 with StackGuard (something close to Cowan's heart, see "StackGuard" in Sec '98). He pointed out that Windows was the first OS to ship with the feature (2004) and now every other major OS does the same. It caused a stir in the audience when Angelos Keromytis said from the back, "OpenBSD came first, everything had SG." "When did OBSD do it?" Cowan asked. "Took time, we had to clean up all the ports. In 2003, version 3.3. So we won by a year and a half," was the response.

"After XP SP2, that's what I thought," replied Cowan unfazed, moving on to described more features of Windows XP Service Pack 2. These included a default firewall, pop-up blocker, and image blocking by default in emails, "which is optional in Thunderbird." A "boo" rose from the audience; "Well, someone should update the Wikipedia page," retorted Cowan to laughter. He also noted other email security features, including Attachment Execution Service (AES), where applications downloaded from the Internet via an email attachment are marked as such. Prompts are raised when a user clicks on the application to execute it. Perry Metzger then asked, "What happens when it gets copied?"

"Not so good," replied Cowan. "But when you copy it, you have to click." To which Sandy Clark replied, "It's kinda like it's not Microsoft's problem?"

"Then users just click to open. Prompt fatigue," sighed Cowan. "Prompts are *not* inherently evil. Prompts that users always say yes to, are a problem." He noted that this is a problem he is actively working on at Microsoft.

Moving on to some browser security features, Cowan described the Windows sandboxing features. He tipped his hat to similar features in the UNIX world, but the PMIE and MIC (Window's sandbox) is on by default. He noted clickjacking defenses: "Don't frame me bro!" Additionally, a SmartScreen blocker for phishing sites and ActiveX filtering for malware. This also caused a stir. Someone from the audience shouted, "Are you now saying ActiveX is secure?"

"No! More secure than it used to be. Security is not a Boolean."

"It's running arbitrary code from the Internet."

"It's not arbitrary, it comes with a certificate!" That received quite a bit of laughter. "If it is ActiveX," continued Cowan, "it is running inside the MIC." He continued to note that plug-ins in Firefox do not use a MIC, and that the SmartScreen has blocked 1 billion malware downloads. He showed a graph to this effect, which also caused a stir.

Cowan deflected comments left and right as he finished his talk. He discussed how Windows undercut the "run as administrator" culture via the UAC, and the number of apps requiring privileged access went from 900,000 to 180,000.

In Windows 7, Microsoft Essentials provided key antivirus protection. AppLock ensured that users weren't using out-of-date and vulnerable apps using a flexible policy. BitLocker was included for full drive encryption, and the virtual accounts feature allowed for per-application user accounts (something UNIX has had since the late '80's).

Cowan concluded by acknowledging that UNIX had a very large security lead, and this was because Microsoft wasn't really trying that hard; Windows has closed the gap across the board, but "once the gap is closed, do users really care which was first?" Finally, he noted that Windows is still the big target and where the money is. The number one security benefit of UNIX may very well be its obscurity; however, don't confuse most obscure with most secure.

DETECTION OF NETWORK ATTACKS

Summarized by Prithvi Bisht (pbisht@cs.uic.edu)

- ***BotGrep: Finding P2P Bots with Structured Graph Analysis***
  Shishir Nagaraja, Prateek Mittal, Chi-Yao Hong, Matthew Caesar, and Nikita Borisov, University of Illinois at Urbana-Champaign

Prateek Mittal presented an algorithm to find P2P botnets. He mentioned that botnet sizes are increasing and that botnets are adopting P2P networking. Such networks are often robust, as there is no central node to be found and brought down. In addition, P2P networks use structured layered topology to be robust and scalable. The current detection schemes detect misuse based on the amount of attack traffic or detect anomalies either by noting deviations from a certain threshold or by using clustering algorithms to isolate botnet traffic. The BotGrep system requires constructing a communication graph whose vertices are Internet hosts and edges represent communication between them. The goal is to extract P2P botnet structure from this graph.

The BotGrep system uses traffic monitors at different ISP sites to construct a host-level communication graph. Inputs from misuse detection systems help differentiate between benign and hostile P2P traffic. An inference algorithm then splits this graph into a botnet graph and a background Internet graph. The inference algorithm uses structural properties of P2P networks. Specifically, random walks compare the relative mixing rates of the P2P subgraph and the rest of the communication graph. The subgraph corresponding to structured P2P traffic is expected to have a faster mixing rate than the subgraph corresponding to the rest of the network traffic.

The approach consists of three main steps: (1) a pre-filtering step reduces huge communication graphs into a smaller set of candidate P2P nodes by using short random walks; (2) a key recursive graph partitioning step uses a modified SybilInfer algorithm, with the intuition that for short random walks the state probability mass is homogeneous, to eliminate non-P2P nodes; (3) a validation step uses heuristics, namely graph conductance, entropy comparison, and degree-homogeneity tests, to decide if a partition is P2P and to terminate the iterations. The scheme was tested using synthesized de Bruijn P2P graphs embedded in the Abilene communication graph. The detection rate was over 90%, and false positives were reported to be manageable. The detection rate remained above 90% for LEET-Chord graphs, but the false positive rate increased significantly. In the presence of large background graphs, performance remained unchanged, and false positives were not dependent on the size of background graphs. Mittal concluded by mentioning that graph algorithms can be used to find botnets, inter-ISP cooperation is useful for security, and stealth and robustness seem inversely proportional.

Yip Fong from MSR noted that the experiments were conducted with a single botnet and asked how the system would work if there are nodes from multiple botnets. Mittal responded that BotGrep can handle multiple overlapping communities through clustering. Fong asked if the scheme could conclude that nodes from different botnets belonged to the same botnet based on a small fraction of nodes. Mittal responded that clustering based on edges instead of vertices would take into account nodes that are part of multiple communities, but the experiments were not done on these lines. A researcher noted that there weren't many details on how the graphs were generated for botnets and what they reflected (e.g., in the case of Kademlia). Mittal responded that the experiments only considered P2P nodes of these graphs; that is, in Kademlia the node degree is logarithmic to the size of the network. The same researcher noted that it was not accurate, graphs did not resemble what the Storm botnet would generate, the generated tool was testing against a flawed model; he wondered if it would work with real bot traffic such as Storm. Mittal agreed; the only modeled component was the P2P overlay maintenance traffic. Someone from Columbia University asked about the motivation for using the Markov-based model for detection. Mittal responded that the most common feature of all topologies was that they were extremely fast-mixing and hence the random-walk-based scheme was used. Someone from the University of New Mexico noted that the false positive rate was 0% and there should be more latent botnets. Mittal said that he didn't have a good answer and that views from multiple ISPs might have identified more botnets.

- ***Fast Regular Expression Matching Using Small TCAMs for Network Intrusion Detection and Prevention Systems***
  Chad R. Meiners, Jignesh Patel, Eric Norige, Eric Torng, and Alex X. Liu, Michigan State University

Jignesh Patel presented a fast regular expression matching scheme using Ternary Content Addressable Memory (TCAM) for Intrusion Detection Systems (IDS)/Intrusion Prevention Systems (IPS). The problem was to quickly scan a packet payload to see if it matched a given regular expression (RE). Existing techniques based on software (use ASIC chips) or hardware (use FPGA) were unsuitable for fast RE updates. TCAM can have three values: 0, 1, *(don't

care), and search is conducted with the content instead of addresses as in traditional memory. The idea is to search against all entries in TCAM in parallel and return the first match. However, the key problem is that the basic implementation produces large TCAM tables. Two optimizations were presented to reduce the space bloat because of transitions. The first optimization exploits the fact that many transitions from one state have common destinations. All such common transitions are merged using the bit-weaving algorithm by Meiners et al. The second optimization reduces common transitions across states. To do that, it reassigns state IDs that are unique to avoid matching unrelated states. However, the optimization needs to retain ordering of states, which is addressed by using the D2FA algorithm by Kumar et al. Patel also mentioned that D2FA has the problem of long chains of deferments which the TCAM-based approach avoids if it finds the deferred state in a single lookup.

Further, space optimization was achieved by consolidating the TCAM tables. This optimization is based on an observation that even after the previous optimizations, some transition tables share common entries although destination states are different. The key idea is to merge multiple transition tables into one table. The two main challenges are: (1) how to merge k tables; (2) which states should be consolidated. The former is addressed by local state consolidation and the latter with the global state consolidation, which uses graph matching and dynamic programming. After minimization, Patel presented a variable-striding algorithm to improve the throughput. To avoid state space explosion, a solution based on k-var-stride DFA (deterministic finite automaton) that consumed 1–k characters was proposed. This led to a linear increase in space. Experiments were conducted with 8 regular expression sets. With transition sharing, the approach generated TCAM entries in the range 1.18 to 2.07 for each state. This was reduced to .32 to 1.17 for each state, below the fundamental limit of 1 entry per state. The highest throughput was approximately 18.58 Gbps. Patel concluded by highlighting that this is the first TCAM-based RE-matching algorithm.

Niels Provos from Google asked whether regular expressions from snort rules were used. Patel responded that 3 of the 8 RE sets in current experiments were from snort, and the work was being extended to cover the rest of them. Had they tried REs used for backtracking? The current focus was on the snort rule set, and backtracking-based REs could not be expressed by DFAs. For handling such REs, DFAs could be annotated with some counting mechanism or scratch memory.

- ***Searching the Searchers with SearchAudit***
  John P. John, University of Washington and Microsoft Research Silicon Valley; Fang Yu and Yinglian Xie, Microsoft Research Silicon Valley; Martín Abadi, Microsoft Research Silicon Valley and University of California, Santa Cruz; Arvind Krishnamurthy, University of Washington

John presented SearchAudit, a tool to leverage search engine audit logs for security analysis. Attackers can craft malicious queries to find misconfigured or vulnerable servers such as the DataLifeEngine server, which was vulnerable to Remote File Inclusion (RFI) and was found with the search term "Powered by DataLife Engine." The idea here was to audit search logs of search engines to understand attack behavior and possibly detect new attacks, and use it for case studies. SearchAudit starts with a seed set of 500 known malicious queries that were taken from underground forums. The seed set was expanded by including queries issued from same IP addresses and finding other queries by issuers of known malicious queries. As attackers use variants of malicious queries, queries in the expanded set were generalized. The generalization consisted of creating regular expressions from each query that captured the essence of the query. This process was repeated as a fixed-point computation.

John discussed validating the outcome of the queries using statistical techniques, e.g., links clicked in results. Queries found by SearchAudit showed significant differences when compared to normal queries in the search logs. John also discussed three sets of case studies. The first case study aimed at early detection of vulnerable servers by analyzing queries that search for vulnerabilities. Such detection can be confirmed by identifying vulnerable servers that subsequently appear in blacklisted domains. The findings indicated that 5% of the identified servers appeared in blacklists subsequently and 12% may be vulnerable to SQL injection. The second case study analyzed queries that search for forums to spam. Findings indicated that some IP addresses sent a huge number of queries, and these findings were consistent with HoneyNet Project findings. The last case study focused on queries for exploiting MSN messenger and found 400 common domains that generated this traffic. Behavior of compromised accounts was analyzed through IM logs and found to be deviating from the normal Messenger logins that typically originated from fewer than four different subnets. John concluded that search queries can provide early indications of attacks and help detect and prevent attacks at an early stage.

Someone asked if the regular expression generation was automated. John confirmed that it was automated and scalable. Niels Provos of Google asked how the expansion of the seed set would work in the presence of churn (people switching IP address) and computers used to make regular queries, and how that would expand to getting false positives. John responded that the expansion was on a per-day basis, assuming that the DHCP changes on less or more than a day's granularity. Further, he referred to the paper

for a technique to detect and eliminate proxies to reduce the false positives.

INVITED TALK

- ***Docile No More: The Tussle to Redefine the Internet***
  James Lewis, Senior Fellow and Program Director at the Center for Strategic and International Studies

  Summarized by Ronnie Garduño (koko@rpg-free.com)

James Lewis focused on the increasing global tension between various countries and the US and its allies over the control of the Internet. The Internet is a mostly decentralized network, but ICANN, the Internet Corporation for Assigned Names and Numbers, does make certain decisions that guide Internet development. The governments of some countries, such as Brazil, China, India, and Russia, charge that ICANN has too much power over the Internet and that it is controlled mainly by the US government. These governments have been working on their policies for controlling the Internet in their various countries for some time now, and they have come to the conclusion that the Internet as it now exists is too open and destabilizing, and that there should be more government control over it. Their attempts to make changes in line with these views are leading to the possibility of a fragmented Internet, with each country's populace able to communicate only with others within that country's fragment of the Internet.

Lewis argued that conceiving of the Internet as a "global commons" ignores the true reality of the situation, which is that the Internet exists due to physical connections and servers, each of which falls under the sovereign control of one nation or another. Lewis further argued that no other country on Earth sees the Internet as a global commons, and suggests instead the idea of a global condominium, within which the citizens of each nation dwell in a shared space with few rules. In this model, each country feels that the Internet in their country is "theirs," not the "world's," leading to the feeling that their sovereignty should also extend to the telecommunications within their borders.

One arena in which this struggle for dominance is likely to take place is in the ongoing standardization efforts. These standards are usually written with intentions to be open, simple, and flexible, goals that do not satisfy those seeking greater control over the Internet. A large part of the problem other countries have with US control over the Internet is due to a disagreement over the extent of the control the US government has over companies within its borders. Lewis disclosed that he has spoken to individuals within governments outside the US who do not believe that the official US government stance on freedom of speech in the media and the operation of companies is an accurate description of the reality of the situation. These people (and probably their respective governments) feel that it would be unrealistic to believe that these activities go on in the US without intervention, and thus they postulate that the US government has a similar level of control to that which exists in various other countries, such as China.

Some other countries claim that the US is a hegemony, a cultural, financial, political, and military force of leadership over the world. Likewise, other countries tend to feel that the US is a kind of controlling power which insists upon assimilating others into its power structures. Some of their fears are economic. There are governments that feel that global organizations like ICANN, the WTO, etc., are part of an overarching strategy to ensure economic dominance on the part of the US. These governments are very interested in the Internet as a tool for economic expansion, but they dislike its political effects and feel that their sovereign powers should extend to the Internet in a way that is currently not entirely feasible, given the Internet's current architecture and control mechanisms. Lewis shared a quote from a Chinese government worker: "Twitter is an American plot to destabilize Iran."

One major problem with a fragmented Internet is that it may not work as well as the current Internet setup. There are two factors held in tension in many countries: the government wants to connect globally for commerce, for research, and for education, but wants to disconnect from the rest of the globe when it comes to politics. The US has mostly left this issue alone, trusting in the strength of current alliances, technologies, and social and economic forces to keep the Internet the way it is. While this is the case, some people are looking to the US for guidance on the future of the Internet, a step which is slowly and quietly taking shape. Lewis argued that this is a necessary step, and says that if the US does not step up to shape the Internet's future, other countries will, and probably in a way which will displease many parties globally. This is interesting in light of a recent poll cited which suggests that the global community has decided that open access to information is a fundamental human right.

Someone asked if foreign governments view American companies developing privacy-enhancing technologies for use outside of the US as part of a coordinated US effort to subvert their control. Lewis pointed out that Hillary Clinton's speech commenting on the Google-China censorship issue implied that we are supporting Google, tarring their reputation even if they are not directly government-supported. Foreign governments don't need total control and don't mind a few people evading their technologies control, but don't like the idea of everyone being able to do it. They also don't understand that American companies are not always working in direct cooperation with the government, since that is not the case in most other countries in the world.

Someone else asked how foreign governments view things like Wikileaks, which don't seem to be controlled by the US government. These other governments often view things like this as proof that the US government is no longer competent. To them, it is evidence of the failure of independent freedoms to ensure social stability, and it reinforces their

need for political control over the flow of information and the Internet.

Another person wondered about countries that are allies of the US, such as Australia and the UK, embracing Internet censorship; how will the US make the case to countries like China to embrace the open flow of information? Lewis replied that he hadn't heard a thing about that yet, although he has been waiting for it. Australia's problem is that they were open about what they were doing. They would probably have done fine if they had kept it a secret, although that may be unrealistic in their case. Many Western European countries are looking at similar implementations of censorship technology, and some of them have already set such systems up, facilitated by the close relationships between the governments of these countries and the telecommunications companies in them. This means that we have already faced accusations of hypocrisy, along the lines of "If you guys can do it, why can't we?" on the issue of censorship. The usual response to these allegations is that the Western world does not generally engage in political censorship or industrial espionage, unlike some of the other countries involved, like Russia. That would be the case the Western world would have to present, that censorship for some limited purposes is acceptable, but not broad-scale political censorship. The technologies to control the Internet have been developed, and countries are realizing that they can extend their control into the domain of the Internet in their boundaries. It is a complex issue because many of these technologies can be used to secure networks, but many people worry that they will be used for censorship. This may mean that we are at a disadvantage as far as cybersecurity goes, since we cannot reasonably implement such technologies.

Someone asked about the future of Net neutrality in the United States. Is the government going to step in and establish rules, or are the corporations going to set up their own? Lewis said he sees the general trend as being that governments are taking a larger role in the control of the Internet. In many countries, this may not be desirable, but it is the case. In the United States, the situation is more complicated because of conflicting interests in Congress. These interests are about evenly matched, so in the end there may be no action taken at all. The telecommunications corporations are insisting that they need to set up their own rules to recoup the expenses of setting of their networks. They also complain that many new Internet technologies and applications are expanding use to the point of straining the networks. A specific example is that of AT&T's 3G network; this network is under constant strain by AT&T's exclusive iPhone and iPad users, many of whom are streaming a great deal. The balance, then, is between return on investment and openness of information flow. Many countries may come to their decisions on this process more quickly than the US, but the messy American political process may prove to be a boon in this case, by allowing enough time for the debate to be fully engaged in by the country at large.

Finally, someone asked about the situation in the United Arab Emirates, in which they recently banned the use of BlackBerry smartphones due to concerns over encryption. Lewis said he knew exactly what their concerns are, in this case, because the US faced those issues more than a decade ago. For a while, the US policy was to restrict the export of encryption technologies, to enable the monitoring of communications from overseas for security and law enforcement reasons. The problem was that these restrictions were unenforceable given the Internet, and many were getting around them by downloading freeware. Some of that freeware was even secretly sponsored by other countries and had back doors, allowing the governments of those countries access to the encrypted communications sent by users of that software. Another problem was an economic concern: encrypted communications are vital to e-commerce, and without the ability to freely use encryption technologies, American companies would have problems expanding their businesses overseas. In the face of all these problems, the US finally relented and removed their restrictions on the export of encryption technologies. The real question, then, is: how long can the UAE keep up their current policy of encryption control, in the face of these security and economic factors? Even though they are a small, oil-rich country, they can't do so for long, given the difficulties involved.

RUMP SESSION

Summarized by Cody Cutler (ccutler@cs.utah.edu)

- ***A Methodology for Empirical Analysis of Permission-Based Security Models***
  David Barrera, Carleton University

Proposing a new methodology for analyzing how permission-based systems are used in practice, David Barrera et al. designed and implemented an algorithm that takes applications as input. It then determines which permissions they require and generates 2D "unique fingerprints" describing this particular application.

- ***Revisiting the Computation Practicality of Private Information Retrieval***
  Femi Olumofin and Ian Goldberg, University of Waterloo

Femi Olumofin and Ian Goldberg question the results of previous work concerning Private Information Retrieval (PIR): "No conclusion is as efficient as the trivial PIR scheme" in practice for multi-server PIR schemes. The hunch paid off: they discovered that the response times of other schemes are one to three orders of magnitude smaller than the trivial scheme, assuming that realistic computational power and network bandwidths are available.

- ***Somnolescent Cryptanalysis***
  Aniket Kate, University of Waterloo

Aniket Kate blazed a new trail in the security world and pioneered what will no doubt be the most effective brute force technique: brute forcing with Cobb from the movie *Inception*. All that is needed is to descend into the dream-world five or six layers deep, where a few real-life minutes will turn into hundreds of millions of years—plenty of time to brute force the target encryption key. Work is currently being hindered by the search to find Cobb himself, which so far has been unsuccessful.

- ***Security on Memory Deduplication***
  Kuniyasu Suzaki, AIST, Japan

Virtual machine monitors can share identical memory pages between virtual machines, just as an operating system shares identical pages between processes. Kuniyasu Suzaki pointed out that memory peeking can infer information about processes running on other VMs on the same physical system. It can be observed that another VM (but it is unknown *which* VM) shares a page by carefully writing to certain pages and watching for timing latencies which signify the virtual machine monitor had to perform a page copy because of copy-on-write.

- ***RFID-Based Electronic Voting: What Could Possibly Go Wrong?***
  Yossi Orren

Although election procedures used to count votes in Israel were perhaps old-fashioned, they produced excellent results. With participation well above 90% and disqualified votes less than 8%, it is a wonder why it was decided to change to an electronic system. Yossi Orren demonstrated several attacks against the new electronic system, ranging from "unsophisticated" to "slightly sophisticated" where he was able to completely erase all previous votes (thus disqualifying the region) and to change the votes for the already cast ballots to an arbitrary candidate.

- ***Dispatch Loops as Execution Signatures***
  Nathan Taylor, University of British Columbia

Nathan Taylor developed a tool that would watch binary execution, find its main loop, and summarize what changes occur in its address space. His tests on fairly straightforward programs turned out well, and he is now interested in using it on sub-programs and slightly trickier executables. Perhaps someday the tool will be able to analyze malware.

- ***What Is the Name of My Cat?***
  Bart Preneel, Katholieke Universiteit Leuven

Bart Corneal studied secret questions used for password recovery techniques. His data, collected when he was asked to vet questions for an online project, shows that 12% of security questions are clever, 54% are simple with low entropy, and 6% are very strange. He asked that everyone help him in this experiment: you can contribute by sending him your security question and answer so he can continue this interesting research.

- ***Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars***
  Srdjan Capkun, ETH Zurich

Srdjan Capkun's new car came with a great new feature: using RFID, the car would unlock automatically if you stood within range for a few moments. He demonstrated, though, that if you don't keep your new key in a special, cool-looking aluminum bag you are vulnerable to a relay attack where someone can open your car even if you are a long distance away from it (http://eprint.iacr.org/2010/332.pdf).

- ***Got Privacy?***
  Maritza Johnson, Columbia University

Maritza Johnson is doing a study concerning privacy: are Facebook users able to configure their privacy controls in a way that actually reflects their intents? You can help her by visiting: http://apps.facebook.com/gotprivacy/.

- ***The Case for Open Source Software***
  Jose Fernandez, Polytechnique Montréal

Jose Fernandez delivered a truly beautiful metaphor relating open source to the monks who tried desperately to integrate strawberries into their foreign lands. However, the birds (evil vendors) made it very difficult for the poor monks. The birds like small strawberries, and cast their seeds far and wide. The monks, like most people, prefer larger, sweeter strawberries, but with the birds "'dropping'" seeds everywhere, the monks needed a special environment to breed their larger strawberries.

- ***NoTamper: Automatic Blackbox Detection of Parameter Tampering Opportunities in Web Applications***
  Prithvi Bisht, University of Illinois, Chicago

Prithvi Bisht demonstrated attacks against some online shopping centers concerning client-side verification; he was able to get the shopping cart to pay for itself by having negative quantities of some of the items! You can read more at http://www.cs.uic.edu/~pbisht.

- ***Simple IPSec***
  Steve Bellovin, Columbia University

Because "95% of options are completely irrelevant to 95% of all users" in IPSec configuration files, Dr. Bellovin developed Simple VPN. It makes the right choices for you automatically—his configuration file is 11 lines. You can grab this tool at http://sourceforge.net/projects/simple-vpn.

- ***The Human-Centered Authentication Attack***
  David Harmon, Columbia University

David Harmon led a study where they explored just how safe our passwords in our heads really are. They found that 1 of 10 users will divulge their password if asked nicely while 8 out of 10 will reveal it if they are waterboarded or encouraged in a similar way.

- ***Secure Systems Cannot Be Engineered***
  Anil Somayaji, Carleton University

For a system to be truly secure, it must have a Roman guard. Unfortunately for us, that requires simply way too much infrastructure for the Internet and all our systems. Roman guards do not scale to the Internet.

- ***Pac-Man on the Sequoia AVC-Edge Voting Machine***
  Alex Halderman, University of Michigan

Alex Halderman and colleagues prove that our faith in electronic voting machines is well justified—they installed FreeDOS on a Sequoia AVC-Edge for the purpose of playing Pac-Man (in celebration of its 30th anniversary) in a matter of hours. Not only did they have complete access to the machine's internals after opening it (without ruining the seal), they were also able to correctly guess which pins on the motherboard to jump in order to defeat the 30-second watchguard timer.

- ***The Word***
  Dan Wallach, Rice University

Dan Wallach closed the rump session with a takeoff on Stephen Colbert's "The Word." He reminds us that we shouldn't be too concerned about elections. Making a secure and usable voting system that preserves privacy is indeed quite hard, so let us not even worry about it—everything will be taken care of for us by hard-working politicians who are genuinely concerned for our welfare. The best policy is "'out of sight, out of mind.'"

DISSECTING BUGS

Summarized by Manuel Egele (megele@cs.ucsb.edu)

- ***Toward Automated Detection of Logic Vulnerabilities in Web Applications***
  Viktoria Felmetsger, Ludovico Cavedon, Christopher Kruegel, and Giovanni Vigna, University of California, Santa Barbara

Felmetsger started her presentation by saying that Web applications are omnipresent and come in many different forms. Common vulnerabilities, such as missing input validation or cross-site scripting, can be detected with taint analysis. In contrast, application-specific vulnerabilities are more difficult to detect, and so far have experienced little attention from the research community.

The prevalence of such vulnerabilities is hard to estimate because there is no specification of what constitutes a "'logic vulnerability.'" Information about such vulnerabilities is scattered across different categories, if they are reported at all. Felmetsger showed four examples of application-specific vulnerabilities in Twitter, Facebook (2), and one in myphile that became public only the day before the presentation. Subsequently, Felmetsger presented Waler, a fully automatic approach based on dynamic analysis and model checking to find logic vulnerabilities in servlet-based Web applications. Waler derives an approximation of a program specification by exercising the Web application with "normal" input. Daikon is used to generate likely invariants on the recorded program execution paths. However, many likely invariants found that way do not represent real invariants. Therefore, to assess the validity of an invariant, Waler employs model checking over symbolic input based on Java pathfinder.

Waler is able to detect two different kinds of vulnerabilities: missed checks on a program path, where an invariant supports a check but a different path leading to the same state does not perform such a check, and inconsistencies between state or session variables and database fields. Since Waler has to take all possible entry points to the Web application into account, the accumulated state becomes too big to handle. Therefore, the authors included different techniques that allow them to detect and prune equivalent states. The authors evaluated Waler on four real-world applications and eight applications that were created by software engineering students as lab assignments. The presentation then elaborated on one of the found vulnerabilities, where a missing check led to unauthorized access with administrative privileges in one of the real-world applications.

In her remarks on future work, Felmetsger mentioned recent progress in the development of Waler, such as support for the Struts framework, and their plans for experimenting with new heuristics to further reduce the state space.

The first question was geared at finding out how Waler finds all possible entry points to a Web applications. According to Felmetsger, all the entry points can be extracted from the configuration file. Eric Eide (University of Utah) was wondering whether the same approach could be applied to application domains other than Web applications, and whether the used heuristics would have to be adapted. Felmetsger agreed that some of the heuristics are Web-application specific, whereas others, such as the heuristics that check for a supporting check on a program path, should be applicable also to stand-alone applications. Eide continued by arguing that 300,000 states are not that many for a model-checking approach, and wondered whether Waler is simply keeping too much state. To which Felmetsger replied that the limiting factor was time, as a simple application consisting of only hundreds of lines of code took a very long time to analyze, and that the effort to analyze a big application (e.g., based on the Struts framework) is orders of magnitude higher.

- ***Baaz: A System for Detecting Access Control Misconfigurations***
  Tathagata Das, Ranjita Bhagwan, and Prasad Naldurg, Microsoft Research India

Das presented Baaz as a solution for detecting access control misconfigurations. He stated three properties that such a system has to provide. It should be preventive, should not require a formal specification or documentation of the access control policy, and should provide high performance. Baaz is built around these design goals as an auditing tool to find potential misconfigurations by checking for inconsistent policies. The system provides the desired perfor-

mance by employing scalable algorithms. Das then continued by elaborating on the two classes of misconfigurations Baaz can detect. Security misconfigurations indicate that a user has access to a resource to which she should not have access. Accessibility misconfigurations signify that a user does not have access to a resource to which she should have access.

Baaz does not require a documented security policy. Instead, it relies on a reference data set that needs to be specified as a binary matrix. This reference data set is used as a proxy for a missing security policy. Baaz checks, for each subject, whether it can detect inconsistencies with the reference data set. Das then presented an example for the matrix reduction in Baaz where the basic assumption is that members in a group should have access to the same resources. The presentation then elaborated on a misconfiguration that was detected by Baaz.

The evaluation of Baaz was performed on three different data sets. The presentation then covered the results of the file server data set in detail, where 18 misconfigurations were detected. To assess the ground truth, the authors spent two days manually examining the data set. Das said that the most time-consuming step was the matrix reduction step, whose runtime grows linearly with the size of the matrix.

Someone raised the concern that the access is controlled by a resource owner and asked whether Baaz required input from the owners, such as having synchronized user names. Das replied that Baaz can be run across administrative domains, assuming that the binary matrix is already available. It is not the intent of Baaz to create this matrix.

- ***Cling: A Memory Allocator to Mitigate Dangling Pointers***
  Periklis Akritidis, Niometrics, Singapore, and University of Cambridge, UK

Dangling pointer vulnerabilities, such as use after free(), are just as dangerous as buffer overflows. Thus they developed Cling as a drop-in replacement for malloc() and new. Similarly to existing approaches, Cling trades memory space for security. Akritidis then presented an example of a vulnerability that Cling is designed to prevent, and two alternatives to prevent such vulnerabilities. The naive solution is never to free any memory, whereas Cling takes the more sophisticated approach, pooling memory and only reusing it for objects of the same type. Cling considers two objects to be the same type if they got allocated by the same instruction (i.e., the address of the call to malloc).

They faced some challenges in implementing Cling. Cling needs to unwind the stack for functions that wrap malloc calls and create different types of objects. Furthermore, as Cling is designed as a drop-in replacement for malloc and new; it does not work out of the box with custom memory allocators. The presentation also included some remarks about limitations of the proposed approach, such as stack-allocated objects and the limited address space on 32-bit architectures. Cling was evaluated on 18 benchmarks by preloading the modified allocator via LD_PRELOAD. The evaluation was performed with regard to memory size and execution time. Furthermore, Cling was evaluated with Firefox, where the requested memory size is almost the same as with the regular system allocator, and only the virtual memory size slightly increased compared to the unmodified version.

Weidong Cui (Microsoft Research) mentioned that memory reusing schemes other than the naive are possible as well (e.g., round robin). Akritidis acknowledged that other schemes exist, but everything that made use of virtual memory so far incurred significant overhead (up to 700%), due to system calls. Furthermore, he said that Cling is designed for production systems, thus protecting the applications, and not for detection purposes, as in other approaches. Robert Watson (University of Cambridge) asked how Cling would behave with C runtime allocations, and closures in particular, where the C library allocates the memory. Akritidis elaborated that the strdup function in the C library exposes exactly this behavior, and Cling treats this function as a wrapper function and is able to resolve the issue by unwinding the stack.

INVITED TALK

- ***Staying Safe on the Web Yesterday, Today, and Tomorrow***
  Sid Stamm, Security & Privacy Nut at Mozilla

  Summarized by Tamara Denning (tdenning@cs.washington.edu)

Sid Stamm began by summarizing the original security tools used by Mozilla: community bug reporting and JavaScript fuzzing. He then described some of the security strategies currently used by Mozilla, as well as strategies that are underway or being considered. Current methods include offering bounties for reporting major security vulnerabilities and fuzz-testing more aspects of the browser. In general, Mozilla is focused on building the browser as a protective agent for the user. Current and contemplated security features include wrapping different browser components, putting plug-ins in their own processes, letting sites specify normal behavior, building in defenses against some CSS attacks, and improving UI indicators of security and trust.

In the future, Mozilla expects a larger attack surface, as the browser harnesses more of a computer's capabilities. The Jetpack system is intended to facilitate add-on security by providing a more compartmentalized system of privileges and APIs. Other goals of interest include a multi-process architecture, an account manager to make a more cohesive registration and login system across sites, better security and trust visualization, something along the lines of a reputation system that reports on the privacy practices of Web sites, reducing browser entropy to increase user anonymity, and associating cookie identity with both its source domain and the page of the domain in which it is displayed.

Many of the audience questions related to either a more effective UI for conveying security and privacy informa-

tion, reputation systems for scoring Web sites, or content security policies. One question addressed Mozilla's stance on the possibility of law enforcement interest in their sync functionality. Stamm replied that the client-side encryption is meant to avoid this kind of scenario. An audience member asked about the security risks associated with a browser account manager. While there are risks, they are outweighed by the security benefits offered to users. How would a Mozilla privacy evaluation system succeed where others (such as P3P) did not gain traction? A reputation system would take the necessary workload away from sites. Stamm also said that the private browsing mode needs to be reevaluated and modified with relevant users and use case scenarios in mind. Tiered sandboxing and a simplified, backward-compatible browser experience are two opportunities for site-side and user-side content security policy.

CRYPTOGRAPHY

Summarized by Ben Ransford (ransford@cs.umass.edu)

- ***ZKPDL: A Language-Based System for Efficient Zero-Knowledge Proofs and Electronic Cash***
  Sarah Meiklejohn, University of California, San Diego; C. Chris Erway and Alptekin Küpçü, Brown University; Theodora Hinkle, University of Wisconsin—Madison; Anna Lysyanskaya, Brown University

Sarah Meiklejohn presented the paper on ZKPDL, a new programming language for the design and implementation of zero-knowledge (ZK) cryptographic protocols. The authors observed an "abyss" between the designers and the implementers of cryptographic protocols: theorists have trouble implementing their schemes at all, and programmers have difficulty implementing these schemes correctly, efficiently, and flexibly. ZKPDL is designed to serve as a lingua franca for both groups, offering theorists a way to express ZK schemes in a concise, familiar way and offering implementers a simple mechanism for incorporating these schemes into their applications.

The authors' cryptography group at Brown had struggled for several months to build an e-cash library for use in a P2P application. The result was messy and difficult to modify, which made changing the details of the underlying ZK scheme difficult. They realized that the ZK scheme could be cleanly separated from their application and reasoned that cryptographers could, if presented with the right interface, code ZK schemes themselves. Their system, ZKPDL, presents both ZK parties—a prover and a verifier—with an identical interpreter and a plaintext ZKPDL program. The prover's goal is to prove knowledge of some fact without revealing anything new about the fact, and the verifier's job is to check the prover's proof. Each party's interpreter loads the program, performs some optimizations where possible, and executes the part of the program corresponding to its role. Meiklejohn showed the interface between ZKPDL and a greatly simplified e-cash library. She demonstrated that ZKPDL programs map cleanly onto the descriptions that cryptographers write in theoretical papers. To demonstrate the efficiency of ZKPDL, she presented performance figures showing various ZK proofs with and without a caching optimization. The authors have made ZKPDL and their e-cash library available at http://github.com/brownie/cashlib.

Peter Neumann asked whether the authors' e-cash library enabled transactions to be traced when it was necessary to do so. Meiklejohn said that e-cash has a basic property that allows cheaters to be de-anonymized. Jeremy Clark asked whether the authors had implemented elliptic-curve primitives; Meiklejohn answered that they had not. Bart Preneel asked whether ZKPDL is designed to be resistant to timing attacks; Meiklejohn answered that it was not.

- ***P4P: Practical Large-Scale Privacy-Preserving Distributed Computation Robust against Malicious Users***
  Yitao Duan, NetEase Youdao, Beijing, China; John Canny, University of California, Berkeley; Justin Zhan, National Center for the Protection of Financial Infrastructure, South Dakota, USA

Yitao Duan introduced P4P, a framework for privacy-preserving distributed computation that supports data-mining operations by decomposing them into vector additions over small fields. P4P aims to address scalability problems that have troubled previous distributed-computation systems; Duan cited several such systems and claimed that their genericity hobbled their scalability. In particular, Duan remarked that existing systems place computationally intensive public-key operations at essential junctures such as simple arithmetic operations, harming performance. P4P's alternative approach is to support only computations that can be decomposed into sequences of so-called private vector additions. This class of computations includes singular value decomposition (SVD), principal component analysis, and a variety of other statistical tools. Duan claimed a run-time improvement of several orders of magnitude for these problems and said that P4P supports operations on up to millions of users or data items.

P4P focuses on a well-known problem: coaxing a group of participants into computing an aggregate function without revealing any node's inputs to any other node. In P4P, participants execute vector additions over small (32- or 64-bit) fields, a fast operation on modern architectures. Duan called these operations private vector additions and said that they are based on verifiable secret sharing. Duan presented an SVD problem as an example: each participant "owns" a row of a matrix A, and the desired computation is the SVD of A. Duan interfaced the popular ARPACK eigensolver with P4P's private vector addition. Each participant's share of the computation is a short sequence of matrix multiplications. To verify the correctness of participants' computations, the coordinating server asks each participant for a projection of its private vectors onto a server-provided random vector; the participant makes a homomorphic commitment to the projection and provides a zero-knowledge proof that the projection is correct. Duan showed run-time figures for a

variety of operations in P4P over a range of numbers of participants. Comparing P4P again to previous systems, Duan pointed out significant (many orders of magnitude) performance improvements for arithmetic operations and statistical computations. The source code for P4P is available at the P4P homepage at http://bid.berkeley.edu/projects/p4p/.

Aniket Kate asked what protections P4P provides against active attacks on the system by cloud servers running the computations. Duan acknowledged that P4P would lose efficiency in the face of such an attack, but he noted that cloud computing providers have no incentive to disrupt computations themselves and typically offer protection against various types of attacks from outsiders.

- ***SEPIA: Privacy-Preserving Aggregation of Multi-Domain Network Events and Statistics***
  Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas Dimitropoulos, ETH Zurich, Switzerland

Martin Burkhart introduced SEPIA, a system that allows network operators to coordinate defenses against distributed cyberattacks without revealing to each other the identities of their customers or the structure of their networks. Burkhart observed that network providers' dislike of detailed data-sharing has stymied past attempts to address global, coordinated attacks. In SEPIA, each participating network deploys a dedicated SEPIA peer that participates in privacy-preserving computations with other SEPIA peers on other networks. In SEPIA's adversarial model, each network's input data is confidential as long as a majority of these peers are honest. Burkhart echoed Yitao Duan's point that secure multi-party computation frameworks have suffered from speed and scalability problems. SEPIA uses Shamir's secret-sharing scheme but optimizes several important primitives for speed and composes these primitives into protocols designed specifically for aggregation of network statistics and distributed event correlation.

Burkhart pointed out that under a naive implementation of Shamir's secret-sharing scheme, a simple privacy-preserving comparison of two 32-bit IP addresses requires 2592 (81 times 32) distributed multiplications, and each multiplication requires a synchronization round comprising m^2 messages, where m is the number of participants. The authors' novel protocols use parallelization to reduce the number of synchronization rounds; they also apply Fermat's little theorem and use square-and-multiply operations to reduce the number of multiplications for an IP address comparison from 2592 to 34. Burkhart showed an example of distributed anomaly detection in which networks using SEPIA would have received early warning of a Skype outage and assessed privately how much their networks were affected compared to other networks. He suggested some optimizations as future work that would further improve SEPIA's performance. Burkhart finished his talk by claiming that SEPIA makes secure multi-party computation practical for networking applications. SEPIA's Web page is http://www.sepia.ee.ethz.ch/.

In a brief question-and-answer period, Aniket Kate referred to a technique that could reduce exponentiation to two multiplications, and Burkhart thanked him.

INVITED TALK

- ***The Evolution of the Flash Security Model***
  Peleus Uhley, Senior Security Researcher, Adobe

  Summarized by Thomas Moyer (tmmoyer@cse.psu.edu)

Peleus Uhley provided an overview of the Security Product Lifecycle (SPLC) that Adobe uses to develop security for all of their various software platforms, including the near ubiquitous Flash plug-in for browsers. He provided insight into the way in which the Flash security model differs from a stand-alone product, and provided attendees with information regarding Adobe's collaboration with various communities.

Uhley began with a discussion of why it was hard to clearly identify the security model of Flash. Specifically, phrases like "Web browser security model" and the same-origin policy are not clearly and explicitly defined, leading to various interpretations of each. He argued that this has led the Adobe Flash developers to support security features within each browser as each browser develops and evolves. He stressed that Flash, much like other plug-ins, exists within a complex ecosystem. He gave several examples of how this ecosystem has changed over the years, one example being the support of private browsing modes. As each browser added support for private browsing, so did the Flash plug-in.

Next, Uhley talked about the difficulties developers face. Chief among these is the evolution of the users. No longer can developers assume that their users have college degrees. Uhley stated that these problems are not unique to Adobe, but that Adobe Flash has become an increasingly popular target, due to the near-universal deployment of Flash. Adobe has faced several problems with regard to this popularity. Uhley highlighted that even a small percentage of successful attacks on Flash can lead to a large number of exploits, meaning attackers are shifting their focus to Flash, and often these attackers are working in larger and larger groups.

Uhley next described how Adobe's responses have evolved over time, as the attack model has changed, but also as the developers gain more insight into how Flash is being used and what users want to accomplish with Flash. The example Uhley provided described the Flash cross-domain communication policy. Initially, this was introduced to handle cases where Flash content developers assumed that two sub-domains sharing a common suffix (e.g., media.example.org and www.example.org) should be able to communicate, even when the browser treated these as different origins. Uhley indicated that reactions to this policy introduction were mixed, leading to refinements in how the policy was

handled. Uhley then discussed several other enhancements to Flash, such as auto update support, and the addition of socket policies.

Uhley finished his talk with a discussion of where Adobe is headed. He mentioned several projects where Adobe has worked with other industry partners and academia, including WEPAWT and Blitzableiter. Uhley also mentioned the Open Screen project and Adobe's efforts to port Flash to new environments, including mobile devices and televisions. Finally, he described work with OWASP and the publishing of specifications for Adobe file formats.

Several questions were raised at the end of the talk, dealing specifically with Flash security. Perry Metzger complained the Uhley had not described a Flash security model, and after some discussion, agreed to continue offline. Adam Drew of Qualcomm asked about the Flash settings manager, specifically highlighting the fact that the interface was dated and difficult to access. Uhley said that as HTML5 evolves, Adobe will be monitoring how local storage settings are handled and adapt Flash's policies to align with other local storage policies. Dan Boneh asked about several recently reported vulnerabilities, including JIT spraying, and wondered how Adobe was dealing with these issues. Uhley responded that Adobe was currently examining several potential solutions and that he could not discuss any one solution in detail. Finally, several attendees asked about communication between the browser and plug-ins. The first highlighted that it would be helpful for Adobe to provide hooks for introspection in the actual plug-in. Uhley responded that he did not have an answer to such a request at the time. Finally, Helen Wang of MSR asked Uhley about unifying the security policies of all plug-ins and allowing the browser to make security decisions on a global scale, instead of each plug-in implementing its own security policy independently of other plug-ins.

INTERNET SECURITY

Summarized by Zhiqiang Lin (zlin@cs.purdue.edu)

- ***Dude, Where's That IP? Circumventing Measurement-based IP Geolocation***
  Phillipa Gill and Yashar Ganjali, University of Toronto; Bernard Wong, Cornell University; David Lie, University of Toronto

Phillipa Gill began her talk by noting the applications that benefit from geolocating clients, such as online advertising, search engines, and fraud detection. However, geolocated targets have incentive to lie, and current geolocation approaches are susceptible to malicious targets. Then she gave an overview of their contributions. They considered measurement-based geolocation in the presence of an adversary who tries to subvert the techniques into returning a forged result. To this end, they developed two models of adversarial geolocation targets: the first, simple one is the Web client being geolocated, and the second, sophisticated one is the cloud provider, which aims to mislead the geolocation algorithm. They developed two specific attacks based on the two adversary models, and evaluated them on delay and topology-based geolocation.

Next, Gill provided background on geolocation and described two major approaches: a database-based passive approach, which is coarse-grained and slow to update, and a measurement-based geolocation, which leverages several landmark machines with known locations to probe and constrain the geolocation. Gill also showed a delay-based geolocation example to illustrate how measurement-based geolocation works, and then introduced their simple adversary model. In this model, the adversary knows the geolocation algorithm and is able to delay their response to probes. In their sophisticated adversary model, the adversary controls the network the target is located in and constructs network paths to mislead topology-aware geolocation.

Interestingly, in their evaluation, they found that against delay-based techniques the adversary has a clear trade-off between the accuracy and the detectability of an attack. Against the topology-aware techniques, in contrast, they found that more sophisticated topology-aware techniques actually fare worse against an adversary, because these techniques give the adversary more inputs to manipulate through their use of topology and delay information. In their future work, Gill described their eventual goal to develop a provable and practical framework for secure geolocation. One approach is to leverage the existence of a desired location, requiring the target to prove they are in the correct location.

No one asked questions; the session chair, Steve Bellovin, joked that sometimes he does not want to be located.

- ***Idle Port Scanning and Non-interference Analysis of Network Protocol Stacks Using Model Checking***
  Roya Ensafi, Jong Chun Park, Deepak Kapur, and Jedidiah R. Crandall, University of New Mexico

Roya Ensafi began her talk by introducing a peach attack, in which the attacker only climbs the hills (with significant cost) to grab delicious peaches when the peaches in a peach orchard the attacker can see stop disappearing. Similarly, in the port scanning attack, the attacker can also leverage the information from a zombie to infer the status of a victim. That is, the attacker uses side-channel attacks to bounce scans off of a "zombie" host to stealthily scan a victim IP address or infer an IP-based trust relationships between the zombie and victim.

After providing some background, Ensafi presented their techniques. They built a transition system model of a network protocol stack for an attacker, victim, and zombie, and they used model checking to test their model for non-interference properties. Their transition-based network stack model consists of five different hosts (two zombies, two victims, and one attacker). Each host has an IP, three different ports and their status, a TCP RST counter, and SYN cache. The rules in the transition system are (1) the attacker can send any arbitrary sequence of packets; (2) the attacker cannot send packets to the victim with its own return IP;

(3) the attacker never replies to any packet; and (4) zombies and victims reply to packets based on typical Linux or FreeBSD network stack rules. They used SAL as their model checker. Two new methods of idle scans resulted from their modeling effort, based on TCP RST rate limiting and SYN caches, respectively. Their empirical experimental results show that it is possible to scan victims which the attacker is not able to route packets to, meaning that protected networks or ports closed by firewall rules can be scanned. This is not possible with the one currently known method of idle scan in the literature that is based on non-random IPIDs. Through modeling two resources, RST rate limitations and a split SYN cache structure, they also tried to capture the distinction between trusted and untrusted hosts, which will appear in the future design of network protocols. Non-interference for these two resources was verified with symbolic model checking and bounded model checking. They showed that in practice it is possible to infer trust relationships and other IP routing rules between the victim and the zombie.

Someone pointed out that according to the RFC protocol in her second example, when sending SYN-ACK to open port, you will get an RST, because SYN-ACK is out of sequence. Ensafi said their result is from the experiment they did on FreeBSD and Linux, and the OS implementation may not exactly reflect the RFC protocol. Someone else asked why the authors didn't consider Windows and Mac. Ensafi replied that she is a fan of Linux and FreeBSD. A third person speculated whether it is possible to use a VM which has great checkpointing, and inside the VM run a real OS rather than creating the transition and performing model checking. Ensafi answered that the idea is great but may face some new problems, such as security during the transition modeling. Robert Watson asked for recommendations on techniques for isolating different bits in the same cache from each other to further differentiate trusted and untrusted machines. Ensafi replied that it was interesting to think about this. Shawn Hernan asked for thoughts on how an attacker in the real world would locate a suitable zombie. Ensafi answered that she is not clear on how the attacker finds the zombies, but an attacker should have the knowledge to locate them.

- ***Building a Dynamic Reputation System for DNS***
  Manos Antonakakis, Roberto Perdisci, David Dagon, Wenke Lee, and Nick Feamster, Georgia Institute of Technology

Manos Antonakakis presented Notos, a dynamic reputation system for the Domain Name System (DNS). DNS is an essential protocol used by both legitimate Internet applications and cyber attacks. But the problem so far is: (1) malware families utilize a large number of domains for discovering the *up-to-date* C&C address; (2) IP-based blocking technologies have well-known limitations and are very hard to maintain; (3) DNS blacklisting-based technologies cannot keep up with the volume of new domain names used by botnets; and (4) detecting agile botnets cannot be achieved by the current state-of-the-art detection mechanisms. Thus, the authors designed Notos, a dynamic, comprehensive reputation system for DNS.

After briefly describing related work such as passive DNS, IP reputation and blacklisting, and DNS reputation and blacklisting, Antonakakis introduced their techniques. Basically, their techniques use passive DNS query data, and extract from network-based, zone-based, and evidence-based feature vectors. It involves a network modeling step along with two clustering steps: one—coarse-grained—uses the network; the other—fine-grained—uses the zone feature vectors. As such, they are able to characterize unknown domains with known network behaviors (for example, CDN, dynamic DNS, or just popular domains) but also with clusters based upon already labeled domains in close proximity. Their reputation function uses the product of both supervised and unsupervised learning steps to compute a reputation score for a new domain indicative of whether the domain is malicious or legitimate. In their evaluation, they applied Notos in a large ISP's network with DNS traffic from 1.4 million users. Their results show that Notos can identify malicious domains with high accuracy (true positive rate of 96.8%) and low false positive rate (0.38%), and can identify these domains weeks or even months before they appear in public blacklists. Their future work includes targeted detection and combines Notos with spam detection systems for improving accuracy as a primary coarse filter.

Reiner Sailer (IBM) asked about the possibility of an adversary taking advantage of their learning technique. Antonakakis answered that it is impossible, as an adversary cannot evade passive DNS. David Reed was concerned about how the authors label the data, since a classic learning algorithm requires reliable labeling. Antonakakis replied their technique is based on public and private blacklists, which should be trusted. Shawn Hernan first asked about the confidence that Alexa lists are in fact legitimate whitelists from non-malicious domains. Antonakakis answered that the top 500 are definitely true. Hernan's second question concerned whether their heuristics that many domain names pointed to a single IP indicates maliciousness works in an IPv6 world. Antonakakis replied their technique works in IPv6, but they may need to revisit their heuristics. Lucas Ballard asked for thoughts on the case of bad guys affecting domains they do not control. Antonakakis answered that an attacker has to compromise Web servers to achieve their goals.

INVITED TALK

- ***Understanding Scam Victims: Seven Principles for Systems Security***
  Frank Stajano, Senior Lecturer at the University of Cambridge, UK

  Summarized by Thomas Moyer (tmmoyer@cse.psu.edu)

In this talk, Frank Stajano presented work that he has been doing with Paul Wilson, a magician on a television show called "The Real Hustle." In this work, Stajano examines scams that Wilson performs on unsuspecting people. After

the scam is complete, Wilson and his team explain the scam in detail and how human nature has allowed the victim to be scammed. Stajano examines these scams, and in particular the victims, and categorizes the principles used to scam the victim. In outlining these principles he provides insight into how system security engineers can take into account human nature when designing security mechanisms.

Throughout the talk Stajano presented video clips of scams presented in the technical report. After each, he discussed how the principles could be applied to system security. The first scam examined was called the three shells game, where the victim must follow a pea hidden under one of three shells while the con artist moves the shells around. The principles behind this scam include distraction and herd mentality. The distraction principle states that a focused user is distracted from other important things, leaving them vulnerable to attack. The herd principle states that a victim will let their guard down when others around them appear to share the same risks. This applies to multi-user systems, where users are more willing to take risks since there are other users willing to accept the same risks.

The next scam, the lottery scam, illustrates the dishonesty principle, the need and greed principle, and the time principle. In the scam, the victim is asked to buy something for less than face value, and is later told that the value of the object is significantly more than originally thought. In the clip, a lottery ticket is forged that appears to be worth several thousand pounds, but later appears to be worth even more than that. The victim is tricked into giving the con artist money for the ticket, at the original value, thinking that he can cash the ticket in and make a larger profit. The dishonesty principle states that the victim's larceny hooks them into the scam, after which the con artist will use this against them to achieve their goal. The same can be said for placing the victim in an embarrassing situation. The second principle is need and greed, which shows that users are made vulnerable by their desire to fulfill their need or greed. Systems engineers need to be aware of users' needs and work to fulfill them; otherwise an attacker can come along and manipulate the user into thinking that the attacker can fulfill their needs.

Another scam examined was the jewelry shop scam, where the con artists pose as a criminal and a law enforcement official. The criminal tries to purchase a high-value item using counterfeit bills, but the "law enforcement official" steps in and prevents the sale. When gathering the "evidence," the high value item is taken as part of the evidence, along with the fake money. The con artists walk out with the money and the item, successfully completing the scam. The premise behind this is that society trains people to not question authority. The con artist leverages this by posing as a higher-ranking official and getting the victim to do what the con artist wants. Systems engineers need to allow users to challenge authority without the risk of being punished, otherwise users will simply be manipulated by the attackers, since the users think that questioning authority will lead to punishments.

In the last scam, the con artist poses as someone in need of help, such as having a flat tire. The con artist is relying on the kindness of the victim. In the scam, while the good Samaritan is changing the tire, the con artist asks if she can warm up in the victim's car. The victim, trying to be nice, gives the con artist the keys to the car so the con artist can turn the heat on. In reality, the con artist is going to steal the car. The car with the flat tire turns out to be owned by some unsuspecting third party, not involved in the scam. The idea behind this scam is that the con artist takes advantage of people's' kindness, which is how many social engineering attacks on systems occur. The attacker poses as someone in need and hopse that the victim will be kind enough to help the con artist.

The only question asked at the end of the talk was about institutional review boards and how Stajano could perform such experiments, as no IRB would approve such experiments. Stajano responded that his partner, Paul Wilson, was the one actually doing the scams and was not subject to IRB approval. Stajano's role is more analysis after the fact.

REAL-WORLD SECURITY

Summarized by Adam J. Aviv (aviv@cis.upenn.edu)

- ***Scantegrity II Municipal Election at Takoma Park: The First E2E Binding Governmental Election with Ballot Privacy***
  Richard Carback, UMBC CDL; David Chaum; Jeremy Clark, University of Waterloo; John Conway, UMBC CDL; Aleksander Essex, University of Waterloo; Paul S. Herrnson, UMCP CAPC; Travis Mayberry, UMBC CDL; Stefan Popoveniuc; Ronald L. Rivest and Emily Shen, MIT CSAIL; Alan T. Sherman, UMBC CDL; Poorvi L. Vora, GW

Richard Carback presented Scantegrity, a voting system designed such that users can confidentially confirm their vote was counted after the election, along with a verifiable tally. It is the first real-world deployment of such a system.

Scantegrity is an electronic voting system with an optical scan reader. The big difference is "invisible ink." A voter uses a special pen when filling out the ballot, and when she marks an oval, a confirmation number appears. These numbers are different on every ballot, and each ballot has a unique identifier. A user is also provided with a confirmation card, where she can write down her ballot's identifier and the confirmation numbers revealed by her "magic marker." After the votes are counted, a voter can go online and enter her ballot number, revealing the official confirmation numbers, and if there are any differences, the voter can challenge the vote.

The most interesting part of the presentation was when Carback discussed some of the real-world pitfalls and successes of the deployment. Tacoma Park had a turnout of 1,723 voters (a good showing), and the "election ran smoothly."

Only 64 voters checked their votes online, and one complained: it turns out that the magic ink "0" looks a lot like an "8." Voter intent issues also arose. Some voters wrote in a candidate already listed on the ballot and did not mark any bubble; others marked the candidate's bubble and wrote the candidate in. These ballots required a hand count. The team also performed an exit poll, and overall voters responded very well to Scantegrity. In conclusion, Carback claimed, "This stuff is ready to go. We did it!"

Peter Neumann asked about overvotes, and Carback explained that overvotes required hand counting. Someone asked whether someone else could determine who you voted for, and Carback responded that the mix hides this information. Abe Singer wondered how their system handled blind voters, and Carback said Takoma Park fell under federal standards (DC) and thus didn't need to support blind voters. Someone asked about absentee ballots, and Carback said they could have sent them the special pens, but they didn't because of the expense. They did design their own ink for the ballots, fill printer cartridges with the special ink, print ballots, and fill pens with the solution that reveals the hidden numbers in the bubbles that are used to verify votes.

- ***Acoustic Side-Channel Attacks on Printers***
  Michael Backes, Saarland University and Max Planck Institute for Software Systems (MPI-SWS); Markus Dürmuth, Sebastian Gerling, Manfred Pinkal, and Caroline Sporleder, Saarland University

After Michael Backes approached the podium to present his work, the first sound the audience heard was the unmistakable beep and whir of a dot matrix printer, producing a chuckle. Backes then asked, "What was just printed?" The crux of his paper, with co-authors Markus Dürmuth, Manfred Pinkal, and Caroline Sporleder, is to answer that exact question.

Dot matrix printers print documents using one to two rows of needles that strike a page through an ink ribbon, producing dots on the paper that form letters and symbols. Printing different letters produces different sounds, and this was known as early as 1991. However, no one has actually produced an end-to-end attack based on this information. One is likely to ask, "'Aren't dot matrix printers obsolete and not used anymore?" Not true. Backes lists a number of examples where dot matrix printers are the norm, including doctor offices for printing prescriptions. Not mentioned by the presenter, but known to the author of this summary, is that these printers are also shipped with many voting machines to produce verifiable paper trails. The relevance of this attack is broader than one may think.

The attack consists of recording the sounds of the printer as it prints a document. The recording is then passed through a recognition phase to produce a set of initial candidates which are pruned down. Language-based improvements are made using standard Markov modeling and n-grams. The results of the attack are striking: 69% of a message can be recovered without a strong straining corpus, and with a good corpus they saw decoding results as high as 95%. They even performed a real-world attack at a doctor's office (on a fake prescription) and were successful in decoding the document.

Ian Goldberg noted that some printers print left-to-right then come back and print right-to-left. He wondered whether the authors could take advantage of this to improve their attack. Backes replied that other noises they would produce might be helpful. An audience member wondered if this could be used to fingerprint a printer or the language being printed. Backes didn't perform that experiment, but he said that getting the printer model is likely possible and recognizing the language should work well. Another audience member was interested in defenses based on adding additional white noise, such as playing fake printer recordings. Backes acknowledged that this might work, but, again it was not tested. One defense that was tested was placing the printer in a box. It didn't work.

- ***Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study***
  Ishtiaq Rouf, University of South Carolina, Columbia; Rob Miller, Rutgers University; Hossen Mustafa and Travis Taylor, University of South Carolina, Columbia; Sangho Oh, Rutgers University; Wenyuan Xu, University of South Carolina, Columbia; Marco Gruteser, Wade Trappe, and Ivan Seskar, Rutgers University

The last talk of the session was presented by Wenyuan Xu, an assistant professor at the University of South Carolina at Columbia. She began by noting that there are an increasing number of wireless devices in cars, and computers are integrated deeply into the mechanical systems and display units of newer model cars. These systems were not designed with security in mind, and in this work, Xu and her co-authors exploited the tire pressure monitoring system (TPMS) in demonstrating poor security and privacy design.

TPMS are wirelessly connected sensors placed in the rims of the tires. They become active when the car is moving faster than 25 mph, and then report regularly to an electronic control unit (ECU) connected to the dash-board display. Fortunately, Xu had a car with such a system, and the team set out recording the wireless signal used. Immediately it became clear that it was Manchester encoding, and they were able to decipher the packet format in less than half a day.

There was no encryption, and they also recognized that each packet had a unique ID. This implies that a TPMS message can be used to fingerprint a vehicle and driver, and potentially track them as the car moves about. The next logical question is: What is the signal range? While the car is parked, without an amplified antenna, they were able to record packets at a range of 10 meters. With an amplifier, this increased to 40 meters. They even tested it on the highway while the car was in motion, and again the signal was easily recorded. Xu joked that she will only claim to have

tested the car at 70 mph and no faster so as not to admit to exceeding any speed limits.

The team also investigated transmitting false TPMS packets to fool the driver into pulling over. Xu described highway robbers in Italy who set up road blocks, but with TPMS hacking they only need to send a "flat-tire" signal to get the car to pull over. This was a very effective attack. Even if the ECU received eight good "inflated-tire" packets and one bad "flat-tire" packet, the on-screen display would warn of a flat tire.

As fortunate as Xu was in having a car outfitted with TPMS technology, she was unfortunate in that it was the guinea pig in all the experiments. In fact, she sent so many forged "flat-tire" packets to the ECU that her system died. Kevin Fu asked about bringing the car back to the dealer after she had crashed the computer: "What excuse did you give the dealer?" Xu replied that she was quite reluctant to reveal the exact reason for the failure. "Just reset the computer. The hardware is fine. Trust me."

Brian Rosenberg (Qualcomm) suggested that the reason the computer would signal a flat tire even while receiving four "good" tire signals was the risk in not reporting a flat tire is much greater than in reporting a flat tire that is not really flat. Dan Wallach asked if they had talked to manufacturers and Xu said they did, attempting to find out how sensors are associated with a car. All the manufacturers would say is that a dealer must install replacement tires for the system to work.

INVITED TALK

- ***Vulnerable Compliance***
  Dan Geer, In-Q-Tel

  Summarized by Ben Ransford (ransford@cs.umass.edu)

Dan Geer posed a series of provocative questions about the following topic: what should be done when a vulnerability is found in a specification rather than an implementation? When such a vulnerability has been disclosed, how do we detect and repair the systems that implement the specification and therefore exhibit the vulnerability? Should protocol designers assume that security flaws will be found in their work and design accordingly? Geer's talk featured an interlude with guest Marsh Ray and a lively question-and-answer session.

Geer pointed to historical examples of so-called vulnerable compliance. In each case, the system's wide install base prevented vulnerabilities from completely disappearing for a long time. Mistakes in the Kerberos Version 4 protocol, introduced in 1988 and retired 16 years later (but still undoubtedly in use), were the first example of full compliance with a specification begetting a vulnerability, according to Geer. Predictable TCP sequence numbers were proved vulnerable in 1985 but not corrected in an RFC until 1996. The wireless networking protocol WEP was not reviewed by cryptographers, has gaping vulnerabilities, and is still widespread today. Further examples include a recent DNS cache-poisoning vulnerability, vendors' hurried implementations of IKE with Xauth, and the proliferation of the flawed sign-then-encrypt (and vice versa) paradigm. Geer observed that, in many cases, these vulnerabilities went unfixed until the disclosure of a working exploit, sometimes years after the vulnerability disclosure. (See the article on p. 26.)

A lesson Geer offered to protocol designers is that if you produce a reference implementation, designers of compatible or derivative systems are afraid to diverge from it—especially for complex protocols. Geer gave Kerberos version 4, SNMP version 1, and ASN.1 as examples of specifications that contained vulnerabilities but were sufficiently complicated that most implementers simply followed the reference implementations. The problem with such close adherence to reference implementations is the loss of implementation diversity, a key principle in the design of the Internet. If merely complying with a specification requires substantial implementation effort, little room remains for critical thinking about the specification's flaws.

Geer gave the floor to Marsh Ray, a security researcher known for his recent discovery of a flaw in the renegotiation phase of the TLS protocol. Ray related the long history of a flaw in the way Windows forwards login credentials. Windows uses a protocol called NTLM to store and transmit password-based credentials. Ray noted that CVE reports are still being issued today for a trivial man-in-the-middle vulnerability that has plagued versions of Windows since 1996. He showed a matrix representing the vulnerability's attack surface over combinations of protocols that use NTLM and said that there were still many opportunities to exploit the vulnerability. The vendor has begun fixing the problem but has not made the repaired behavior the default, because they do not want to break backward compatibility. Ray drew several lessons from the saga. If breaking backward compatibility is painful, do it once and comprehensively fix the problem. Highly visible attacks that focus on one facet of a system can distract from potentially more severe underlying vulnerabilities. Protocol designers may find their work burned into silicon, complicating repairs. Encrypted communications can hide the existence of underlying flaws or disguise attacks.

Geer discussed remediation strategies used in the past. He gave examples of top-down remediations, such as when AT&T enlisted the help of legislators to punish phreakers while they invested in more secure protocols. A similar approach to vulnerable Internet nodes might treat such a node as an "attractive nuisance" akin to an unprotected backyard pool. Another possible remediation strategy would be for protocol designers to issue an expiration date for new protocols. He mused that inflection points such as Y2K are an appealing juncture for protocol switchovers. He closed his talk by wondering whether sound defensive strategies might

lead implementers away from Postel's famous Robustness Principle: systems should become conservative in what they produce *and* in what they accept.

Ray showed a video (by Liam Schneider) of credential-forwarding attacks on NTLM while the audience responded to the speakers. An audience member pointed out that many countermeasures, such as patches to DNS servers and TCP implementations, import the notion of security into systems that were designed without security in mind, which seems like a mistake. Ray commented that DNSSEC in particular draws attention away from a need to reform the badly broken PKI infrastructure for SSL. Geer noted that we must sometimes deploy imperfect solutions. Wietse Venema asked whether Geer thought SMTP should have been expired; after all, it lacks authentication and authorization. Geer compared email to financial markets, which have taught us that we can build systems that are too complex to operate, and suggested that perhaps SMTP ought to be made modular, with parts that can be swapped out if they are found to be vulnerable.

David Reed pointed out that users have a need to assign blame for security problems, but that standards committees cannot simply be held liable; he remarked that security is a societal process rather than a property. Ray and Geer agreed; Geer suggested that organizations should allocate resources to deal with security failures. Another audience member wondered why there are no insurance groups that estimate the costs of security failures; Geer said that insurance is enormously complicated but that some organizations have been thinking about it, but only to insure users against failures rather than insure designers against flaws. An engineer in the audience noted that civil engineers, for example, can be held accountable for flaws in the physical artifacts they design (e.g., bridges), but noted that a comparable notion of accountability on the decentralized Internet would unduly hinder development. Geer offered the maxim "Freedom, security, convenience—choose two," and suggested that perhaps the people who deploy, rather than design, systems should be held accountable for failures. As the allotted time drew to a close, David Reed pointed out that although every software company warns against using their software in critical systems, most, with a smile and a wink, upsell their wares into just such systems.

POSTER SESSION & HAPPY HOUR

Summarized by Sandra Rueda (ruedarod@cse.psu.edu) and Rick Carback (rick.carback@gmail.com)

[Editor's note: This session was so popular that it wasn't possible to interview all the poster presenters: it was too noisy, and having good food and drinks made things more difficult. Nevertheless, it was a lot of fun. We are just sorry that not all poster presentations could be covered.]

- ***GuardRails: A (Nearly) Painless Solution to Insecure Web Applications***
  Jonathan Burket, Patrick Mutchler, Michael Weaver, and Muzzammil Zaveri, University of Virginia

GuardRails is a Ruby security tool. It is designed to help developers avoid common Web application security vulnerabilities using annotations instead of explicit security checking code. It provides support for data input sanitization and access controls, and also avoids information disclosure vulnerabilities when access denied errors occur.

- ***Tools for Tracking and Understanding Keyword-Based Internet Censorship***
  Antonio M. Espinoza, Ronald J. Garduño, Leif A. Guillermo, Veronika Strnadova, and Jedidiah R. Crandall, University of New Mexico

This is a probe for detecting words and phrases that trigger Chinese Internet censorship actions. It uses character similarities, named entity extraction, and latent semantic analysis to create the list of censored topics. It can potentially be used as a data source for the Concept Doppler system (ConceptDoppler.org).

- ***Advancing the Science of Cyber Security Experimentation and Test***
  Jelena Mirkovic, USC Information Sciences Institute

DETER is a project that aims to provide a public repository of experiments in the area of computer and network security for educators and students to analyze in related college courses. The experiments include exercises about intrusion attacks and detection, firewall management, spoofing, forensics, denial-of-service attacks, and worm behavior. More information at: http://www.isi.edu/deter/.

- ***MedVault: Health Professional Access to Source-Verifiable Patient-Centric PHR Repository.***
  Mustaque Ahamad, Douglas Blough, Ling Liu, David Bauer, Apurva Mohan, Daisuke Mashima, Bhuvan Bamba, Balaji Palanisamy, Ramkumar Krishnan, Italo Dacosta, and Ketan Kalgaonkar, Georgia Institute of Technology.

MedVault is a project to develop new techniques for the storage, maintenance, and sharing of health records while protecting such records from unauthorized use and disclosure. MedVault uses Merkle hash trees to provide minimal disclosure of information and integrity verification at the same time. The patient only needs to authorize the release of a specific piece of data and the hash codes associated with the remaining branches of the tree for the reader to be able to verify the integrity of the data. MedVault also uses attribute-based policies to release information. Attribute-based policies enable patients to make fine-grained decisions about data sharing.

- ***Redacting PHI in Neurological Images using XNAT***
  Alex Barclay, Laureate Institute for Brain Research and Institute of Bioinformatics and Computational Biology, University

of Tulsa; Nakeisha Schimke and John Hale, Institute of Bioinformatics and Computational Biology, University of Tulsa

XNAT is an open source platform designed to handle medical imaging and data. XNAT uses the DICOM standard (Digital Imaging and Communications in Medicine standard) for handling medical images. The problem is that the DICOM standard includes Protected Health Information (PHI), that is, information that can be used to identify an individual. Furthermore, the image itself may include information that can be used to identify an individual. This poster highlights the need for a tool to redact the entire PHI data stack, including DICOM headers, text, and the image byte stream, to ensure privacy of the data.

- ***Embedded Firmware Diversity for Smart Electric Meters***
  Stephen McLaughlin, Dmitry Podkuiko, Adam Delozier, Sergei Miadzvezhanka, and Patrick McDaniel, Pennsylvania State University

Current smart meters belong to a category that is known as monoculture, meaning that a large percentage of these meters have the same hardware and software. From a security point of view, monocultures represent a high risk since attacks that succeed on one of the elements can be repeated on all of them without much additional effort. Traditionally, software diversity techniques have been used to mitigate attacks on monocultures. However, the techniques that can be used on smart meters are limited because of the hardware requirements associated with many of them and the hardware limitations of the smart meters.

This poster presents redundant address encryption to provide "lightweight control flow integrity" to prevent random errors after an exploit attempt. Redundant encryption using different keys to protect return addresses provides reasonable guarantees to protect the smart meters.

- ***Process Firewalls: Mechanism and Utility***
  Hayawardh Vijayakumar, Sandra Rueda, Divya Muthukumaran, Joshua Schiffman, and Trent Jaeger, Pennsylvania State University

Current operating systems support access control policies at the granularity of a program and cannot enforce finer-grained access control policies. Therefore, an operating system's ability to enforce a policy depends on what interface a program is using to access a given OS resource. This project proposes a Process Firewall mechanism to enforce policies with a finer granularity that would allow access based on what interface a program is using to access a given OS resource. This behavior is analogous to a regular firewall's behavior that enforces policies for a given host based on network features such as a port number.

- ***Graph Cuts Can Be Used to Solve Security Problems***
  Divya Muthukumaran, Dave King, and Trent Jaeger, Pennsylvania State University

This poster proposes that security problems arising from information flow errors can be modeled as a graph cut problem. A cut solution to the graph cut problem is a solution for the security problem. This kind of problem includes mediation placement in programs (placement of declassifiers and endorsers), privilege separation (since we want to split the code), and policy error resolution (errors indicate illegal information flows and thus a cut suggests where to mediate the flow). The challenges to model the problem include identifying sources and sinks, and converting cuts to the appropriate security solutions.

- ***Securing End-to-End Provenance: A Systems and Storage Perspective***
  Kevin Butler, University of Oregon; Patrick McDaniel, Stephen McLaughlin, and Devin Pohly, Pennsylvania State University; Radu Sion and Erez Zadok, Stony Brook University; Marianne Winslett, University of Illinois

This paper presents a mechanism, Kells, that enables a USB device to evaluate the integrity of the host it is being connected to, before releasing any of the information it stores.

Since Kells can identify the machine that it is plugged into, it is possible to build a provenance chain at the block level based on reads and writes from a given machine. Once the host is validated by the device, it can be considered to be within the TCB, so requests are trusted. At the block level there is no concept of users per se, but the device can consider users through other means, such as biometrics on the USB drive.

- ***Verifying Cloud Integrity: Making the Cloud Do the Dirty Work***
  Joshua Schiffman, Thomas Moyer, Hayawardh Vijayakumar, Trent Jaeger, and Patrick McDaniel, Pennsylvania State University

This work addresses two questions: (1) How do we ensure the integrity of the results produced in a cloud environment? (2) How can customers verify integrity?

This project designed and implemented a cloud verifier (CV) to answer these questions. The cloud verifier is a component in the cloud that can verify the integrity of the virtual machine monitors (VMMs) in the cloud. It does so based on an integrity criterion that is shared with the customers. Customers decide if the CV criterion meets their own. The CV also provides an IPSec key that customers can use to establish trusted sessions with their own VMs (for instance, to send keys to access encrypted data stored in the cloud). This key is generated by a VMM for the VMs it is hosting. Since the CV has verified the VMM's integrity, it signs the key and sends it to the customer.

- ***tNAC: Trusted Network Access Control***
  Ingo Bente, Josef von Helden, and Joerg Vieweg, Trust@FHH Research Group, University of Applied Sciences and Arts, Germany; Marian Jungbauer and Norbert Pohlmann, Institute for Internet Security, University of Applied Sciences, Germany

The tNAC project aims to develop a trustworthy Network Access Control solution. tNAC builds upon Turaya, the

secure operating platform, and TNC@FHH, the open source Trusted Network Connect implementation. tNAC ensures that by integrating the capabilities of Turaya, which are rooted in the Trusted Platform Module, and TNC@FHH, which gathers security relevant information about each endpoint, only those endpoints that match the policy of the provider will be allowed to access the network. Endpoints that try to lie about their current security state will be detected. For further information about tNAC, please visit www.tnac-project.org.

- ***Moving from Logical Sharing of Guest OS to Physical Sharing of Deduplication on Virtual Machine***
  Kuniyasu Suzaki, Toshiki Yagi, Kengo Iijima, Nguyen Anh Quynh, and Cyrille Artho, National Institute of Advanced Industrial Science and Technology; Yoshihito Watanebe, Alpha Systems, Inc.

This is a proposal to use memory- and storage-deduplication to increase security. Application binaries are translated by pseudo-static converter (for example, "statifier" in Linux). The binaries share necessary libraries and prevent search path replacement attack, GOT (Global Offset Table) overwrite attack, Dependency Hell, etc. They require more storage and memory, but deduplication techniques reduce the increase.

WEB SECURITY

Summarized by Manuel Egele (megele@cs.ucsb.edu)

- ***VEX: Vetting Browser Extensions for Security ulnerabilities***
  Sruthi Bandhakavi, Samuel T. King, P. Madhusudan, and Marianne Winslett, University of Illinois at Urbana-Champaign

  Awarded Best Paper!

Firefox currently has around 25% market share, and 150 million plug-ins (i.e., extensions) are in use. Firefox extensions are written in JavaScript and executed in the same context as the chrome, the browser's frame and controls. Extensions run as part of the browser and thus have access to everything you do with your browser. After giving a brief overview of the current submit process for Firefox extensions and its weaknesses, Sruthi Bandhakavi elaborated on the idea and the threat model behind VEX.

Extensions are assumed to be benign and vulnerabilities to be the effects of buggy extension code. Vulnerabilities can be exploited by a malicious Web site. To protect from these threats, VEX employs static analysis to check for explicit information flows that bridge the two trust domains for JavaScript in the Firefox browser: the chrome and content contexts.

VEX identified a vulnerability in an RSS reader extension. Bandhakavi prepared a demo exploit to attack this vulnerability and demonstrated the effects. VEX uses abstract heap data structures for objects, methods, and properties to compute precise flows between objects. Currently, VEX contains three different flow patterns, and the authors were able to identify six vulnerabilities in 2452 extensions they analyzed with VEX.

The presentation concluded with a glance at future work: Bandhakavi said that they want to study and classify known vulnerabilities, and employ a constraint solver to improve VEX. The project Web site can be found at http://www.cs.illinois.edu/~sbandha2/VEX/.

Peter Neumann asked about the limitations of the employed flow analysis and how we can get out of the unfortunate situation that we have untrusted operating systems, browsers, and browser plug-ins. Bandhakavi answered that the limitations for static analysis apply to VEX too. However, VEX was designed as a bug finding tool and thus is not able to state the absence of bugs. More effort should be put into designing languages that can be analyzed reliably. Someone asked about false positive and false negative evaluation and where the ground truth comes from. Bandhakavi replied that VEX did not detect all known vulnerabilities. For example eval constructs still pose a limitation to VEX. Two undergrads worked to systematically create attacks employing fuzzing techniques, but it was really tough to create such a tool, because each extension is unique in what inputs it accepts. She emphasized the need for tools like VEX that could at least point to the presence of an attackable flow in order to test the extensions manually. The flows detected in the extensions could eventually turn out to be not attackable for various reasons outlined in the paper and therefore become false positives.

Collin Jackson (CMU) wondered how many extensions loaded content that got executed in the chrome context from HTTPS-secured URLs instead of regular HTTP. He asked why one would ever allow content from nonsecure sources to be passed to the eval statement. Bandhakavi felt that only allowing HTTPS sources might be too restrictive.

Helen Wang asked how VEX compared to inline monitor approaches that are built into the browser. Bandhakavi clarified that VEX is intended to help extension editors to vet extensions before they get approved, and thus is able to find vulnerabilities before they get deployed to the browser.

- ***Securing Script-Based Extensibility in Web Browsers***
  Vladan Djeric and Ashvin Goel, University of Toronto

Vladan Djeric presented their work to provide protection against privilege escalation vulnerabilities in script-based browser plug-ins. Djeric started his presentation with a brief overview of the Firefox architecture. One of their design principles was to implement their approach with no modification to existing extensions. Djeric then divided existing vulnerabilities into three classes: code compilation vulnerabilities, luring vulnerabilities, and reference leaks. The threat model assumes benign extensions and untrusted data being executed as privileged code. They added a dynamic taint propagation engine to the Firefox browser.

Existing security measures in Firefox advocate the use of a taint propagation scheme. For example, name space separation in the browser creates a natural boundary for taint labels, and privileged scripts usually handle untrusted data with care.

Based on the taint propagation scheme, the authors implemented techniques to detect vulnerabilities in all of the three vulnerability classes. The authors implemented and evaluated a prototype of their technique in Firefox 1.0.0. The reason to choose this rather old version is that the published security bulletins are very detailed. Out of 14 advisories their approach was able to detect 13 vulnerabilities. The only vulnerability that was not detected results from an incomplete implementation. More precisely, the authors did not implement the taint propagation throughout the HTML parsing engine. To evaluate the false positive rate, the system was exercised by a human Web surfer for five hours, resulting in one false alert. Also, an automated crawler visited the top 200 Web sites of the Alexa Web site ranking, also resulting in one false positive. The performance evaluation showed slowdowns up to 30% in micro-benchmarks.

Ian Goldberg (University of Waterloo) wondered how the system handles JavaScript closures. According to Djeric, using closures to interact between trusted and untrusted content is not common. Peter Neumann wondered whether their approach could benefit from a more fine-grained interpretation of taint, as opposed to the binary tainted/not-tainted scheme. Djeric responded that he prefers to err on the side of caution. Venkat Venkatakrishnan (University of Illinois, Chicago) compared this work with the previous talk on VEX and asked whether static or dynamic analysis techniques are better suited to protect the user from vulnerable extensions. Djeric stated that their approach does not only detect vulnerabilities in extension but also in the browser itself, if, for example, vulnerable wrappers are present. Sruthi Bandhakavi, the presenter of the previous talk, described a problem with dynamic analysis: once a problem is detected, the user has to make a decision on how to proceed (i.e.., ignore warning and continue or terminate the execution).

David Wagner (University of California, Berkeley) wondered about the methodology that was used to measure the 30% performance impact. Djeric agreed that this slowdown is not negligible but said that it's too little to be perceived in day-to-day browsing. Niels Provos (Google) said that the user cannot trust extensions. Djeric agreed and reiterated that their work was aimed to protect the user from vulnerabilities in benign extensions.

- ***AdJail: Practical Enforcement of Confidentiality and Integrity Policies on Web Advertisements***
  Mike Ter Louw, Karthik Thotta Ganesh, and V.N. Venkatakrishnan, University of Illinois at Chicago

Mike Ter Louw presented AdJail. He introduced a running example of the Yahoo Webmail interface that he would use throughout the presentation and discussed the issue of a context-sensitive ad on Facebook that would fetch the profile pictures of a user's friends and use them in dating service advertisements. The specific example suggested that the user might be advised to date his own wife through this dating service. The presentation continued by stating five design goals for AdJail. These goals are confidentiality and integrity of sensitive page data, a consistent user experience, support for ad scripts that perform contextual advertisement, transparency towards the ad-networks, and support for all major Web browsers.

AdJail creates a shadow page for each real page that contains the unmodified ad script. Access to content of the real page is mediated by two JavaScript components embedded in these two pages, the real and shadow pages. These components employ DOM interposition and are responsible for mediating access and forwarding events. Furthermore, AdJail defines a policy language to annotate read and write properties for certain content areas pertaining to ad scripts. The AdJail prototype was evaluated with six ad networks and was integrated with the Roundcube Webmail application. Their prototype implementation resulted in a slowdown of around 200ms for rendering the advertisements.

Niels Provos (Google) wondered how this approach relates to confidentiality breaches, where ad scripts steal browser history, and how this work relates to Caja. Ter Louw replied that such attacks are outside the scope of their work. Also, Caja has to transform the ad script before it is delivered, which is not necessary for AdJail and is undesirable, as it may raise false positives in ad networks' click-fraud detection mechanisms. Lucas Ballard (Google) asked how Flash advertisements are handled. In AdJail, Flash advertisements cannot interact with JavaScript. Algis Rudys asked whether AdJail allows the publisher to limit the write access to areas where context-sensitive ads will be placed (e.g., an ad should only be able to add content, such as links for keywords, but not be able to rewrite the whole content). Once a region is marked as writeable, the ad can perform any modification to the area, including a complete rewrite.

Matt Jones (Facebook) wondered whether the amount of data that is transmitted to the ad network can be limited, or if an ad script could send the whole email content to the ad network. Ter Louw answered that, commonly, only keywords are extracted and transmitted, but in general it would be hard confining such behavior. The last question was geared at finding out how the ad script and the AdJail scripts communicate and whether a malicious ad script could talk to the AdJail script in the original page directly, bypassing the protection. Ter Louw responded that the

AdJail script on the real page does not trust anything from the shadow page, and all policies are enforced in the AdJail component on the real page.

INVITED TALK

- ***How Cyber Attacks Will Be Used in International Conflicts***
  Scott Borg, Chief Economist, US Cyber Consequences Unit

  Summarized by Sunjeet Singh (sstattla@gmail.com)

Scott Borg, an expert in the area of cyber warfare, assesses cyber security risks to the US and closely studies ongoing cyber conflicts internationally. Borg discussed various recent real-world examples to draw a line between the true potential of cyber attacks and the actual extent to which cyber attacks play out today. He then presented specific statistics that explain the strategic implications of such cyber attacks and said that cyber attacks are set to become the major form of warfare in the future. (During his talk he repeatedly cited his summary on the recent conflict between Russia and Georgia: search for US-CCU-Georgia-Cyber-Campaign-Overview.pdf.

Cyber attacks offer many unique advantages over physical attacks, mainly in that they can be anonymous, highly targeted, overwhelming in impact, and, at the same time, reversible. The reliance of any nation on information technology makes it a prospective target for cyber attack. Apart from the critical infrastructure, many modern weapon systems in use today use IT, and this makes cyber security all the more crucial. Cyber wars have been witnessed at several levels in recent conflicts all over the world, with each successive conflict increasingly sophisticated.

In the recent conflict between Russia and Georgia, there was high strategic coordination between cyber and physical attacks. Although there is no firm evidence that Russia was behind the cyber attacks that took out Georgian government Web sites, media communications, and power infrastructure during that period, all these events were so highly synchronized with on-ground military advances that it seems implausible that a third entity could have been behind the cyber attacks. It is believed that the Russian cyber attackers had control over much more of Georgia's critical infrastructure than they exercised, which would go to show that the attack was highly organized and disciplined. Georgia in turn came up with a counterattack by releasing malware on social networking Web sites using the Russian language, thus targeting Russian users. The suffering of Georgia from this war has left behind bitter traces in the minds of Georgian people, which suggests to Borg that future attacks might not be as controlled as the Russian attack was.

In attacks less controlled than Russia's, it is likely that a local conflict could lead to a global impact. In a recent staged government experiment, hackers were able to seize control of a US power grid generator and caused it to self-destruct. Having established that critical infrastructure can be attacked and that physical damage can be inflicted by cyber attacks, it is reasonable to assume that for higher-value targets such as pipelines and refineries, the damage would be severe. For example, a disruption in electronic supply or oil and gas chains in Asia would cause global repercussions.

Given the potential impact, unlike many specialists in this field who believe that cyber warfare will supplement conventional warfare and act merely as a force-multiplier, Borg argued that cyber techniques will govern physical techniques to become the major weapon in future. The purpose of any war is to establish control over the adversary, and cyber warfare provides the means to do it in an effective manner.

At this point, the audience had questions on the practicality of large-scale cyber attacks, e.g., on a nation's complete power grid, on how well such attacks can be controlled, and on how asymmetric the attacking and defending sides can be. To these, Borg's reply was that the whole world is high-tech today. Low-launch attacks from minimal infrastructure and from any part of the world can potentially cause great impact. Although it is not easy to take advantage of an attack in a controlled fashion, it is much easier to inject malware to cause damage.

SECURING SYSTEMS

Summarized by Andres Molina-Markham (amolina@cs.umass.edu)

- ***Realization of RF Distance Bounding***
  Kasper Bonne Rasmussen and Srdjan Capkun, ETH Zurich

Kasper Rasmussen presented a way to realize a distance bounding protocol using RF communication. Distance bounding protocols are run between two entities, the verifier and the prover. The prover's goal is to prove to the verifier, using a challenge response protocol, that he is within a given physical distance from the verifier. More precisely, in a model where the verifier is trusted and the prover is untrusted, the prover cannot pretend to be closer than he really is. Also, after the protocol is run, the verifier has proof that the prover is within a certain distance.

Rasmussen noted that robustness in a distance bounding protocol comes from requiring that an attacker must take essentially zero processing time to respond to challenges. The authors propose the use of Challenge Reflection with Channel Selection (CRCS) in distance bounding protocols instead of bounding protocols that require the prover to interpret the received bit before replying to it. Not only is interpreting unnecessary, but it is the reason why alternatives are slow. Rasmussen explained that even alternatives that implement this using XOR are inadequate, not because XOR itself is slow, but because protocols require that full symbols be received before processing them, and receiving a

symbol can take microseconds. The fastest known approach relying on XOR has a processing time of 300 ns, which translates into an error in distance bounding of 50 meters. In contrast, the proposed solution that uses CRCS is well suited for distance bounding because it does not require the interpretation of the bit received before replying. This allows the prover to receive, process, and send a challenge in less than one nanosecond, which translates into an error in distance bounding of about 15 centimeters.

The main idea of this approach is that, using two channels, the prover reflects a challenge back to the verifier without interpreting it. The use of one channel would encode a 1 and using the other would encode a 0. Thus the prover's choice of a channel would encode a bit of knowledge of a nonce. A distance bounding protocol would, in addition, rely on cryptographic signatures and the integrity of a challenge to protect against two attacks, distance fraud and mafia fraud. Rasmussen described a wired implementation and referred interested members of the audience to the paper for a wireless implementation.

Ian Goldberg (University of Waterloo) noted that a prover could collude with an external attacker that is closer to the verifier to prove that the prover is as close as the attacker. Rasmussen responded that in that case the attacker becomes the prover, and it is just a matter of preventing the prover from sharing his credentials. Avishai Wool from Tel Aviv University noted that in the wired implementation described, high frequencies (~3.5 GHz) were used, but that some important applications, e.g., contact-less cards, work at low frequencies (~13 MHz) and with slow processors. He asked if the proposed solution would still apply in these cases. Rasmussen responded that in theory the approach should still be valid but that it would be an engineering challenge to deal with such low frequencies. Another member of the audience asked if the mixer in the proposed approach could be replaced by switching a modulation on and off to encode a bit, for example. Rasmussen responded that indeed other approaches are possible, as long as they avoid symbol processing and interpretation before replying.

- ***The Case for Ubiquitous Transport-Level Encryption***
  Andrea Bittau and Michael Hamburg, Stanford; Mark Handley, UCL; David Mazières and Dan Boneh, Stanford

Andrea Bittau presented tcpcrypt, a TCP extension that would enable end-to-end encryption of TCP traffic by default. He started by listing the three main requirements for a solution that would encrypt the vast majority of TCP traffic: performance, endpoint authentication, and compatibility with existing networks and legacy applications. He then said that no existing solution achieves all three.

Bittau provided examples in which tcpcrypt would improve the security guarantees on sites like CNN, Amazon.com, Facebook, or Bank of America. He hinted that this could be done while also improving overall performance. Next, he listed some advantages and disadvantages of providing security at an application layer with SSL or at the network layer with IPSec. In particular, he mentioned that while IPSec could work with all applications, it could break NAT and would not be able to leverage user authentication. These claims about IPSec would later be challenged by a member of the audience.

The authors claimed that tcpcrypt would provide high server performance by pushing complexity to the clients, would allow applications to authenticate endpoints, and would provide backwards compatibility with all TCP applications, networks, and authentication settings. Performance is achieved because encryption and decryption operations in RSA are not equally expensive. Thus, it is possible to design a protocol in which the cheap operations are on the server side. Doing so would allow servers 36 times better performance than SSL. However, this would require a different approach to authentication, using session IDs. Additionally, tcpcrypt would use existing SSL infrastructures to batch-sign session IDs and thus amortize the cost of RSA operations. In order to provide compatibility, tcpcrypt would modify the initial SYN-TCP with a SYN-CRYPT to probe for tcpcrypt support. If the server ignores the probe, the client would fall back to regular TCP. However, if the server supports tcpcrypt, then both parties would continue with a tcpcrypt negotiation encoded in TCP options.

After going over various protocol and implementation details, Bittau explained that even though better performance can be achieved with tcpcrypt than with SSL, performance gains would vary according to various ways of providing authentication. Bittau referred the audience to http://tcpcrypt.org to obtain a copy of tcpcrypt and install it in their systems. The authors offer tcpcrypt in a Linux kernel implementation and a userspace implementation that runs on Windows, Mac OS, Linux, and FreeBSD. Bittau concluded his talk by demoing tcpcrypt on a Web application that allows clients using tcpcrypt to post messages into the tcpcrypt Hall of Fame.

David Reed pointed out that piggy-backing on the SYN packet may allow DoS attacks. Bittau responded that tcpcrypt is implemented using mini-sockets requiring one single bit, and thus the server state on the SYN is cheap. Another member of the audience said that by using tcpcrypt instead of IPSec, one would lose protection on other transport layer protocols such as UDP or SCTP. He also challenged Bittau's claims about IPSec not being able to provide individual authentication and not being able to play with NAT. Bittau responded that, indeed, the authors had restricted their attention to TCP traffic, which is the majority of the Internet traffic. As for his previous claims, Bittau stood by them and invited the member to continue the discussion offline.

- ***Automatic Generation of Remediation Procedures for Malware Infections***
  Roberto Paleari, Università degli Studi di Milano; Lorenzo Martignoni, Università degli Studi di Udine; Emanuele Passerini,

Università degli Studi di Milano; Drew Davidson and Matt Fredrikson, University of Wisconsin; Jon Giffin, Georgia Institute of Technology; Somesh Jha, University of Wisconsin

Lorenzo Martignoni proposed an architecture that can be used to automatically generate procedures to repair a system after it has been infected with malware. Martignoni made the case that preventing an infection is not always feasible and that current malware detection software does not always leave systems in a stable and safe state after repairing them. The authors showed that their approach was able to revert 98% of the activities performed by 200 pieces of malware, in comparison to the 82% achieved by the best leading commercial solution.

Martignoni described the challenges of generating remediation procedures. One complication is that malware code is usually obfuscated and, therefore, hard to analyze. Moreover, the behavior of this type of software is typically non-deterministic, and remediation usually takes place only after an infection has been detected, so the previous state of the system is not completely known. Next, Martignoni described their approach, which consists of three steps: (1) they construct "infection relations" by extracting generalized patterns of clusters on behavior graphs obtained by running the malware in diverse controlled systems; (2) infection relations are then used to construct remediation procedures; and (3) these procedures are performed in the infected system to revert the effects described by the infection relations. The major limitation of this approach is that attackers could increase the behavior generalization of their malware, thereby decreasing the ability for this system to obtain complete results. Also, only a subset of modified resources can be properly restored. In particular, deleted files or user files cannot be restored.

Katsunari Yoshioka (Yokohama National University) asked about the malware samples used for the evaluation part of the paper. Katsunari explained that in his experience these are hard to analyze because they are often not self-contained, so parts of their code may be obtained from remote locations. Martignoni agreed and added that, in fact, some pieces of malware may simply crash and stop working. However, these were not considered in the paper.

INVITED TALK

- ***Grid, PhD: Smart Grid, Cyber Security, and the Future of Keeping the Lights On***
  Kelly Ziegler, Chief Operating Officer, National Board of Information Security Examiners

  Summarized by Leif Guillermo (laag@unm.edu)

Kelly Ziegler explained that the talk would be kept at a high level for a policy perspective and would explain how the electric grid works and how the smart grid came to be. She hoped this would provide a useful background for understanding some of the issues we are facing now related to cyber security and other security-related issues. She also mentioned that at the end of the talk she would speak about the regulatory framework surrounding the power grid.

There were three main areas of focus on the power grid: power generation, transmission, and distribution. There are roughly 5000 power plants with roughly 160,000 miles of power lines distributed over one million square miles. The North American power grid can be broken down into three interconnections. These interconnections are described as the eastern connection, the western connection, and ERCOT, which is located in Texas. These connections can be thought of as the largest machines in the world, because they are all synchronized.

Between supply and demand, energy output must meet energy demand at every instant. There are three main energy supplies, and a supplementary supply. For the base load of energy demand—large consumers of electricity such as factories and commerce—coal power is generally used. The intermediate energy load requires gas units. Finally, for peak loads and supplementary supply, natural gas is used. Peak loads generally occur between 3:00 and 6:00 p.m.

There is an imaginary barrier known as the "Chinese Wall" which separates bulk power system policies from distribution policies. The bulk policies are regulated at the federal level, and these policies deal with power plants and power generation, whereas distribution is regulated at the state level and deals with how power is transferred to consumers. Due to this barrier, regulating demand can be problematic. The smart grid is a temporary solution to gain control over demand. It implements a variety of solutions: automatic meter reading, distribution automation and generation, demand response, and supervisory control and data acquisition (SCADA) control systems and sensing. Automatic meter reading was the first temporary solution, deployed earliest on major industrial and commercial locations. This method provided a more detailed hourly and time of use billing. It also helped to cut down on the number of meter readers and allowed for various different configurations depending on the needs of the user. Distribution and transmission system automation allows operators greater control and management. Distributed generation allows for smaller generating units to serve the energy load locally. Demand response is a technique to flatten out the peak of energy consumption. One implementation of this method is for people to opt to reduce their basic energy rate, but when it's really hot outside and the energy load is very high, the rate is increased.

In the 1990s, deregulation occurred in the electricity sector, which continues to allow people to trade electricity. Since then, there have been huge amounts of growth in the energy sector. Eighty-five percent of relays are now digital. Originally security was not a big design requirement for the power grid, but after the terrorist attacks of September 11, 2001, we started realizing that security is actually a big issue for us. Since the original design of the grids wasn't

implemented with security integration in mind, the issue of security has become a very troublesome obstacle to tackle.

The greatest threat is the potential for an attacker to attack multiple key nodes on a system. Both physical security and cyber security are enormous issues, and new vulnerabilities arise all the time. Before addressing security, however, many business issues need to be addressed in order to be sure that the security issues are feasible. Managing the risk of implementing security measures seems to be the most important piece in keeping the power grids safe.

There are nine critical infrastructure protection standards designated by the North American Electric Reliability Corporation (NERC), which reports to the Federal Energy Regulatory Commission. NERC is self-regulatory and is governed by the utility companies. NERC's argument is that if they don't make certain requirements critical, the utilities don't have to comply with those requirements, so this is another roadblock in the way of security. An important idea to take away from the talk is that the current regulatory structures set in place to address cyber security in smart grid technologies are inadequate, in part because of the complexity of the whole smart grid system and the fact that the smart grid wasn't designed with a high level of security in mind.

Many questions focused on attacks that destroyed transformers and on the resiliency of the existing system. Evo Dismet pointed out that destroying a substation could take out a city for a year. Ziegler responded that taking out three or four substations would cut off DC from power. Some substations use very large custom-designed transformers and take over 18 months to build. Cathy Jenks of Sun/Oracle asked if the US has agreements with the countries who manufacture this equipment, and Ziegler pointed out that Aviva, in France, would likely replace transformers in France before they would help other countries. Jessica Smith from MITRE wondered about communication and load balancing, asking if it made sense to connect the east and west networks. Ziegler said that it didn't, although it had been considered for better use of renewables. But things are quite reliable as they are, and connecting the two networks might create more unreliability.

Steve McLaughlin of Penn State asked about spare equipment at substations to prevent cascading failures. Storm restoration is something utilities do all the time. But transformers are hugely heavy and very difficult to move around, although mobile transformers do exist. But if a cyber attack occurred that took out many nodes, recovery could take years.

USING HUMANS

Summarized by Femi Olumofin (fgolumof@cs.uwaterloo.ca)

- ***Re: CAPTCHAs—Understanding CAPTCHA-Solving Services in an Economic Context***
  Marti Motoyama, Kirill Levchenko, Chris Kanich, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage, University of California, San Diego

Marti Motoyama began this talk by describing the goal of the paper, which is to evaluate CAPTCHAs as a security mechanism by looking at CAPTCHAs-solving ecosystems. CAPTCHAs, or Reverse Turing tests, are first-line defense mechanisms against large-scale, automated exploitation of Web resources. In their most common form, CAPTCHAs consist of alphanumeric characters distorted in some ways and are presented as visual challenges to the user. CAPTCHAs are easily solved by humans, are easily generated and automated, but are hard to solve by computers.

To help attackers circumvent the defenses posed by CAPTCHAs on targeted Web sites, commercial CAPTCHA-solving services have emerged consisting of automated software solvers and third-party human solving services. Some of the identified limitations for software solvers are the requirement for skilled programmers, difficulties in achieving high accuracy, and the ease with which defenders (i.e., designers of CAPTCHAs) can adapt and defeat solving algorithms using better obfuscation of their CAPTCHA challenges. Marti said that it does not make sense to invest in software solvers. Even the popular Xrumer solver has been adapted recently to leverage human-based CAPTCHA-solving services.

The solving market is globalized because of several factors, including cheap Internet access, the commodity nature of CAPTCHAs nowadays, and the non-specialized skill requirements for solving CAPTCHAs. It is easy for these service providers to aggregate on-demand CAPTCHA-solving requests and outsource them to workers recruited from some of the lowest-paid labor markets around the world. Many of these services are able to solve CAPTCHAs for retail prices as low as $1 per thousand. Wholesale and retail prices are declining in this demand-limited market.

In this study, the authors tried to understand the security of CAPTCHAs by asking economics questions that compare the cost of solving CAPTCHAs, using either of the two approaches, to the cost of the assets that CAPTCHAs protect. Essentially, CAPTCHAs add friction to the business models of attackers and should be evaluated in terms of how efficiently they can undermine attackers' profitability. Some of the findings from the study were validated in an interaction with the owner of a successful CAPTCHA-solving service.

Stephen Jenbecky from MITRE suggested the use of culturally dependent CAPTCHAs, such as ones that pose visual challenges that depend on a geographical area. Such

CAPTCHAs can help reduce the effectiveness of foreign human laborers used by CAPTCHA-solving services. Jeremy Epstein (SRI International) asked how many times Klingon (Star Trek) CAPTCHAs were tried, and Motoyama said 222. Epstein commented that human solvers could learn from examples. Cody Cutler asked about the legitimacy of CAPTCHA-solving services, and whether or not such services pay their workers. Motoyama said they did pay their workers.

- ***Chipping Away at Censorship Firewalls with User-Generated Content***
  Sam Burnett, Nick Feamster, and Santosh Vempala, Georgia Tech

Sam Burnett described Internet censorship as a global problem not limited to oppressive regimes alone but including democratic governments as well. Existing solutions to defeat censorship and surveillance of network communications rely on helpers (e.g., proxies) to relay communications between users in a censored regime and those outside the censored regime. Commonly used anti-censorship systems, such as Tor, have three shortcomings. First, it is easy for censors to block proxies if the proxy list is public. Second, a user in a censored regime can often not deny participating in a communication. Third, the success of such systems relies on the benevolence of volunteers outside the censored regime to provide a network of proxies (i.e., requires dedicated infrastructure).

Burnett called their solution Collage, which is a method for bypassing censorship firewalls by hiding messages inside user-generated content such as photos on Flickr, tweets on Twitter, and videos on YouTube. The vast amounts of user-generated content on many Web sites provides an unlimited amount of cover traffic that makes it difficult for censors to block all possible sources (i.e., no dedicated infrastructure to block). Burnett said that they have developed tools to store censored data in user-generated content using such techniques as steganography and watermarking. Unlike Tor, where a user is easily implicated by merely contacting a relay, Collage provides its users with some level of deniability, since they can hide their messages inside harmless-looking messages (e.g., photos, videos, etc.).

Sending a message with Collage requires the sender to obtain the message and pick a message identifier for the message, which should only be known to the intended recipient. Then the sender obtains cover media such as personal photos and embeds the message in the cover media. Next, the sender uploads the user-generated content to some hosts. The receiver can then find and download the user-generated content from the hosts and extract the message from it. Embedding a message into cover media consists of two steps: (1) encrypt the message with the message identifier; (2) split the ciphertext into many chunks using erasure coding. Each erasure-encoded chunk corresponds to a task, and the ciphertext can be reconstructed from any k-subset (i.e., offers robustness). Another problem addressed is how message receivers can identify the locations of message vectors without having to crawl the entire user-generated content on a host, and without any immediate communication with senders. Their solution was to use task mapping to map message identifiers to these locations. Senders publish message vectors so that receivers can get the vectors when they perform tasks. For example, a task may be for the receiver to search YouTube or Flickr with a particular keyword.

The performance metrics for Collage include sender and receiver traffic overhead, sender and receiver transfer time, and the storage required on content hosts. These metrics vary a lot depending on the content host and type of tasks that receivers need to perform in order to retrieve message vectors. Burnett described a case study on sending a news article and covert tweets using Flickr and Twitter as content hosts. The message sizes were 30KB and 140 bytes, receiving times were two minutes and half a minute, and storage needed on hosts 600KB and 4KB, respectively. Sam also ran a demo of a Collage application, which is available for download at http://gtnoise.net/collage.

The presentation ended with highlights of some areas for further research, such as statistical deniability against traffic analysis, learning timing behavior from users, and Tor bridge discovery.

- ***Fighting Coercion Attacks in Key Generation using Skin Conductance***
  Payas Gupta and Debin Gao, Singapore Management University

Payas Gupta began this talk by saying that many techniques have been proposed to generate strong cryptographic keys. While some of these techniques—biometrics, for example—possess desirable security properties such as ease of use, unforgettability, unforgeability, and high entropy of the keys, they cannot resist coercion attacks. In this attack, the adversary forces the user to reveal the key. The focus is on finding ways that would make the user incapable of generating correct keys when he or she is coerced. They assumed that the adversary knows about the coercion-resistant property; otherwise the user's inability to generate a correct key might be interpreted as stubbornness, and that could endanger the life of the user.

Gupta described their proposed solution to achieve coercion resistance, which is to incorporate users' emotional status or arousal state, through the measure of skin conductance, into the process of key generation. They extended a previously proposed biometric key generation technique that relies on voice, to use both voice and an emotional response parameter of the user's skin conductance. Key generation follows a look-up approach based on the original biometric key generation technique. Their reason for choosing skin conductance over other physiological signals (e.g., heart rate, skin temperature) was because skin conductance is cheap to measure and the deviation in measurements is small.

Gupta described a user study to evaluate their solution consisting of 39 participants (22 male and 17 female) who were undergraduate and graduate students aged between 18

and 30. They ran two experiments to capture the emotional response of participants, using skin conductance sensors attached to their fingers, when they were in a calm condition and when they were stressed. Each user generated a cryptographic key in each state. The approach used to stress participants was by showing them a frightening horror movie. The result of the study shows that different cryptographic keys were generated for the two experiments and the approach has moderate false positive and false negative rates.

Someone asked whether the authors obtained internal ethics approval before conducting the user study. Gupta confirmed that they did. The same person was concerned about why they had to put participants in such a high-stress situation and questioned the validity of their result because many variables might be going into the result without them knowing. Another person commented that skin conductance might depend on the climate of the room where the person is located. The same person said that skin conductance is a measure of stress, which may be unrelated to whether or not a person is coerced. Lucas Ballard (Google) commented that sometimes it might be difficult to detect why authentication failed even in a non-stressed situation, due to high variability in biometric measurements (i.e., voice and/or skin conductance). The failure of either or both of these could be due to other factors such as noise in the environment, illness, or tiredness.

INVITED TALK

- ***End-to-End Arguments: The Internet and Beyond***
  David P. Reed, MIT Media Laboratory

  Summarized by Joshua Schiffman (jschiffm@cse.psu.edu)

David Reed started his talk by providing a historical background that led to the publishing of his original End-to-End (E2E) argument paper, which he notes is one of the most cited papers and least understood ideas. Originally, Reed and his advisor Saltzer had been collecting design principles from security experts from the NSA and IBM, but stressed that no one understood computer security at that time. In 1976, he shifted his focus to networking protocols and how they could be factored into layers, which itself generated much argument as to which features should go into each layer. Reed mentioned an early paper of Clark's, "The Design Philosophy of the DARPA Internet Protocols," which stressed the technique of multiplexing *existing* interconnected networks as a major design goal. Another paper Reed and Clark published, "An Introduction to Local Area Networks," also emphasized that a technological innovation is utilized in two stages. In the first, the innovation is used to improve the performance of what was already being done; the second stage is the discovery of new applications not conceived of beforehand. Finally, Saltzer, Clark, and Reed published the E2E paper, which identified a non-intuitive structure of some systems and presented an argument of what not to put in the core of the communication network.

Reed defined the argument abstractly and then said, more concretely, that secure message delivery can only be done at the endpoints, despite what networking companies tout as a secure network. He then said that a major confusion point is deciding what constitutes a function F and what constitutes an endpoint. Some examples include traffic management and capacity reservation, which could be done entirely in the network. F is a quality, property, or attribute of the network that is emergent, but not necessarily a property of all the parts. Security and reliability were identified as emergent because a system may be reliable despite an individual piece being insecure or unreliable. Reed believes that the E2E argument should really have been called End-to-End Argumentation, to carefully define such functions and avoid confusing them with techniques that designers want in their networks or products.

The talk then moved to some earlier publications that picked up the E2E idea. One notable example was Lessig's article in *The New Republic* that placed the E2E argument into a legal domain and introduced new concepts like "network neutrality." Reed also described how the E2E argument was similar to the financial theory term, Real Options, which suggests one should delay making decisions that limit options, thus preserving those options for the future. He then noted that this introduces a design trade-off of preserving options versus optimizing. Leaving a system unoptimized introduces uncertainty, but is not a problem if it is built into the design. Security for example, deals with uncertainty as much as it does threats.

Reed then touched on some of the controversies around the E2E argument. In *The Future of the Internet and How to Stop It*, Zittrain calls for abandoning or modifying E2E arguments if the Internet is to be secure, robust, and safe; E2E lets the users control the Internet, and the unity of the network enables real-time sensing and dissemination of users' information. Reed notes that these arguments have a compelling meaning to them, but they are not compelling enough to change the design principle. In response to Clark and Blumenthal's "Rethinking the Design of the Internet," which says that policy requirements that employ CALEA-like rules and spam blocking are not compatible with E2E, Reed questioned whether the techniques used to address the issue were right in the first place.

In closing, Reed reiterated that design principles survive because they make use of clear systematic reasoning. Such principles are neither gospel nor prime directive, but a pattern to reason by. He repeated that the E2E principle helps to manage uncertainties by dealing with how functions should be implemented and that we should not confuse functions with the techniques, features, or capabilities for achieving that function.

Ron Rivest from MIT noted that the E2E idea presupposes that one can implement things correctly, but most people cannot implement security right. If there are attacks on incorrectly built communication networks, where should the complexity be designed? Reed replied by questioning the wisdom of modularity, saying that we often confuse ideal properties with a module itself or the specification with the chip implementing it. Thus, the problem is a logical issue, by which we map the model to the object, and is not an issue with the E2E principle. Ben Norrik from Google asked Reed to define an endpoint; Reed answered that it is inherent in the design of what you are building.

Another audience member asked what Reed thought of nation states that dislike the Internet's inability to be controlled precisely because that function is not in the network. Reed mentioned that some aspects of the network came from the need for a globally addressable scheme for all participants. What these nations do is form their own private Internets, much as companies create private networks. An attendee pointed out that Reed suggested that security is not something to build into the network and asked whether Reed felt putting ACLs into an OS kernel was a design error. Reed said he disagreed with his co-authors that it was practical to design a secure kernel. They had originally been tasked by the military to build a kernel that functioned like a network, which passes messages from process to process and respected a multi-level security lattice. However, such a kernel was of no military value, because military operators frequently declassify messages in the field and thus break their own requirements to be practical. Ultimately, they learned that the specification was extremely flawed and had they applied the E2E argument to kernels, they would have realized they could not build what they needed into it.

5th USENIX Workshop on Hot Topics in Security (HotSec '10)

August 10, 2010
Washington, DC

CLOUD AND WEB

Summarized by Katherine Gibson (gibsonk@seas.upenn.edu)

- ***Visual Security Policy for the Web***
  Terri Oda and Anil Somayaji, Carleton Computer Security Laboratory

Terri Oda presented ViSP, a visual security policy that builds on previous mash-up work, that she and her co-author hope will address the numerous and diverse ways in which Web site security can fail. Oda pointed out that approximately 83% of Web sites will have a security vulnerability in their lifetime, and that two-thirds have one right now. As an example, a user posting a comment on a Web site may inject code into their comment that would change the login box on that page such that if a user typed in their username and password, this information could then be exploited. As another example, advertisers may want to edit the content of the page on which their ads are displayed, perhaps negatively changing reviews of a competing product. What ViSP aims to do is to prevent attacks like these by isolating elements on a page.

ViSP is based on four tags: a box tag, which defines a region of interest; a channel tag, which is placed within a box and defines a communication channel from another box; a multibox tag, which indicates that all sub-elements should be automatically boxed; and a structure tag, which is necessary for layout but does not have any security properties. The ViSP system can be thought of as "drawing boxes" around volatile content on Web pages, not only to prevent malicious code from affecting other parts of the page, but also to prevent vulnerable areas of the site, such as logins, from being modified without authorization. As Oda succinctly put it, you "don't want sharks in your sandbox." ViSP currently has some limitations—it has no support for isolating elements without a visual representation, and it has no way to specify partial access between boxes, among others—but Oda and her co-author have released it as a JavaScript-based Firefox 3 add-on which seems intuitive to use. Additionally, the visual element of ViSP seems much more in tune with how Web designers think and is much easier to comprehend and implement, while still protecting against a wide array of attacks.

During the discussion, Lucas Ballard (Google) asked how ViSP fits in with CSS, HTML, and JavaScript. In the beta version of ViSP, it goes on after all the other components, but Oda hopes that the final version will be integrated. Ballard also asked how ViSP deals with scripts that lack a visual presence. Oda stated that there is nothing to do about those at the moment, but that a lot of non-visual scripts are tied to a visual element, giving more support for the idea that designers' minds work visually. Collin Jackson (CMU) asked how ViSP could prevent an attacker from "pushing" the boxes off the page. Oda said that ViSP would need to fix the box location to prevent this kind of attack. Finally, Adam Aviv (University of Pennsylvania) asked how ViSP assures the user that it's using the appropriate security policy, and Oda responded that you don't, but that even without ViSP most users will assume Web pages are inherently okay.

- ***Cybercasing the Joint: On the Privacy Implications of Geo-Tagging***
  Gerald Friedland, International Computer Science Institute; Robin Sommer, International Computer Science Institute and Lawrence Berkeley National Laboratory

According to Gerald Friedland, geo-tagging is cool, generates revenue, and helps to organize pictures and videos: there are over 3 million geo-tagged YouTube videos and over 180 million geo-tagged photos uploaded to Flickr. Unfortunately, people are unaware of geo-tagging, possibly

a consequence of most technology requiring the user to opt out of their content being geo-tagged, rather than opting in. Following the example of the site pleaserobme.com, which is drawing attention to the privacy lost by Twitter users when they update on their mobile away from home, the authors performed a few case studies to determine how difficult it is to use geo-tagging to determine information that most people think of as private.

For the first case study, using Twitter and tweeted pictures (which upload your location if you update via a geo-tagging-capable device, such as an iPhone), the authors "stalked" a celebrity, determining where he lived, where he walked his dog, and where his kids went to school. They then took on the For Sale section of Craigslist, using the geo-tagged pictures of the sellers' items to determine the location of the seller, which could be done to such accuracy that the address could be determined. Finally, they used a few fine-tuned search parameters to find the home locations of YouTube users who were on vacation, looking for those who had uploaded videos more than 1,000 km away from their likely home location within the past week. All of these "cybercasing" scenarios were successful for various reasons: many users don't realize that they are releasing geo-tagged information, that fast and easy-to-use APIs can pull out and search through the geo-tagged data, and that the default for geo-tagging is high-precision and enabled. In addition, services such as Google Maps allow geo-tagging information (e.g., longitude and latitude) to be easily correlated with street addresses.

In the discussion, Lucas Ballard (Google) asked Friedland if he had any thoughts on whether changing the APIs would have an effect. Friedland responded that although the high precision currently used is unnecessary, inferring can still provide a lot of information, so it is also necessary to educate users. Stuart Schechter (MS Research) questioned whether geo-tagging really provided better information than what criminals currently have access to. Bill Cheswick (AT&T Labs) half-jokingly suggested changing the location of geo-tagging information to that of the nearest police station. Finally, Adam Aviv (University of Pennsylvania) asked if there were any trends, given the widespread use of iPhones (which have geotagging on images and mobile posts turned on by default), as well as wondering whether useful information could be lost in the noise created by this ubiquitousness. Friedland responded that geo-tagging will only get more common, and that despite the massive amounts of geo-tagging data available, with the use of APIs it is surprisingly easy and fast to sift through photos and tweets.

- ***On the Impossibility of Cryptography Alone for Privacy-Preserving Cloud Computing***
  Marten van Dijk and Ari Juels, RSA Laboratories

Marten van Dijk's talk posits that cryptography alone is not enough to enforce privacy when dealing with cloud computing services, even when one takes into account such powerful tools as fully homomorphic encryption. Instead, van Dijk suggests the use of a nested hierarchy of three classes. The first class is private single-client computing, in which a single client's data is given to the cloud in an encrypted form such that when the cloud performs the requested function, it does not have access to the data. The second class is private multi-client computing, in which multiple clients that do not necessarily trust each other give the cloud their encrypted data, the cloud performs the requested function over all of the data, and the results are given back to the appropriate clients, without the cloud having access to the unencrypted data. The third class is private stateful multi-client computing, which differs from the second class only in that the access control policies are dependent on the full history of data a specific client has sent to the cloud. Essentially, in all of these classes, the cloud can't see unencrypted information from the clients, and the clients can't collude with the cloud to see other clients' information.

Ian Goldberg (University of Waterloo) posited that two-party obfuscation algorithms may be possible. Next, Adrian Perrig (CMU) brought an upcoming paper on how to use cloud computing securely (by Rosario Gennaro, Craig Gentry, and Bryan Parno, titled "Non-Interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers," to be presented at Crypto 2010) to van Dijk's attention. Finally, Lucas Ballard (Google) wanted to know whether the given schemes would work if a user was worried about entropy rather than cryptography. Van Dijk replied that although he was not sure, his intuition told him that it would be very difficult.

SYSTEMS AND DEFENSES

Summarized by Rik Farrow (rik@usenix.org)

- ***Popularity Is Everything: A New Approach to Protecting Passwords from Statistical-Guessing Attacks***
  Stuart Schechter and Cormac Herley, Microsoft Research; Michael Mitzenmacher, Harvard University

Stuart Schechter began by quipping that in high school, popularity is everything. He next outlined several threats against passwords by using statistical guessing: first, the password file itself being compromised, using the RockYou loss of 30 million passwords as an example; second, online dictionary attacks using the variant of statistical guessing (most popular first), and using bots in a botnet to attempt guesses to avoid lockout; third, attacks against sites that require special characters in passwords, trying simple substitutions, like "$" for "s." Schechter pointed out that many sites use restrictions on password choices, winding up with passwords like "Pa$$word1" and "blink182" (a name of a popular band that includes numbers).

Their solution is to limit the number of people sharing a given password. Rather than storing passwords, which itself is dangerous, they use four truncated hash tables with count-min structure, similar to a counting Bloom table. When their software gets a password, they hash it, truncate

the hash, look it up in four versions of the hash table, and increment all matching hash buckets. Collisions are actually desirable, as they want false positives to exist so that the lookup cannot be used as an oracle. They can use these hash tables to inform a user that the password they chose is "too popular," thus limiting the fraction of users with the same password. Popular strategies for choosing poor passwords, like including the user name, still must be applied.

Paul van Oorschot (Carleton University) pointed out that up to 100 people, the limit chosen in this work, can still share popular passwords such as "password1". Schechter countered by saying that the worst that can happen is that the attacker can compromise 100 accounts, when Microsoft is protecting many millions. Bill Cheswick (AT&T Labs) asked if they had experimented with usability, and Schechter said that they have asked permission from RockYou for their (already public) data set to seed their hashes, but have not done this yet. Someone wondered about using this for much smaller sites, to which Schechter suggested that small sites could pool their hash tables.

- ***Moving from Logical Sharing of Guest OS to Physical Sharing of Deduplication on Virtual Machine***
  Kuniyasu Suzaki, Toshiki Yagi, Kengo Iijima, Nguyen Anh Quynh, and Cyrille Artho, National Institute of Advanced Industrial Science and Technology; Yoshihito Watanebe, Alpha Systems Inc.

Kuniyasu Suzaki gave an interesting talk about replacing logical sharing of shared objects, such as DLLs and shared libraries, with deduplication of physical memory. Using DLLs and shared libraries can itself be risky, as libraries get searched for during dynamic linking. The week after this paper was presented, exploits against this very feature in Windows versions were made public. Suzaki suggested static linking of files as a defense. He also mentioned an attack against files using ELF format, by changing the Global Offset Table to point to locations of the attacker's choosing. Again, static linking solves this problem. Static linking also solves issues like dependency or DLL hell, and problems with mismatched libraries after package updates.

The disadvantage of using statically linked binary files is that they are much larger in size and require more memory when executed than programs that use shared libraries. The authors' solution is to use memory deduplication. Suzaki pointed out that this is already done in virtual machines, such as VMWare ESX, Satori, and Differential Engine. Their implementation used their own program, statifier, on binaries, and KSM (Kernel Samepage Mapping) for memory. Binary file sizes increased 40 times on average, but less physical memory was required. Also, booting is faster as dynamic reallocation overhead is eliminated. In conclusion, Suzaki said that self-contained binaries strengthened OS security by preventing some attacks, as well as avoiding dependency hell.

Nathan Taylor (University of British Columbia) wondered if the suggested use case, in the cloud or IaaS, was correct, and Suzaki said that their experiments suggest that this is the best use. Taylor then asked if this requires an extra layer in the VM, and Suzaki replied that there is no extra layer, just an extra module. There is a weakness in their approach, one that Suzaki explained during the Rump Session, in that an adversary could detect whether a particular memory image had already been loaded.

- ***Embedded Firmware Diversity for Smart Electric Meters***
  Stephen McLaughlin, Dmitry Podkuiko, Adam Delozier, Sergei Miadzvezhanka, and Patrick McDaniel, Penn State University

Stephen McLaughlin explained that this research began as a penetration test that showed that an attack that works once works everywhere. Smart meters include both limited local processing and a wireless interconnect. A smart meter can report an outage to your house, but can also be used to disconnect your power.

McLaughlin described three security concerns: fraud, that is, hacking meters to reduce the cost of electricity; privacy, as detailed load profiles can be used to infer a lot about the inhabitants of a house; and blackout exploitation, where an attack cuts off power to one or many houses.

Smart meters, so far, are almost a perfect monoculture, with identical hardware and firmware. The current meters use simple processors, with 8-bit registers, no protected mode, and no segments or MMU. Their solution is to use software diversity by encrypting return addresses, using a simple XOR and three different keys. In this scenario, failed attacks will have the side effect of causing the firmware to fail or misbehave. Stephen McLaughlin concluded by asking for suggestions, such as reducing TCB code that needs diversification. He pointed out that 10 million smart meters have already been deployed, with a planned replacement time of 30 years (and a 10-year MTBF).

Someone asked if it was possible to attack back up the chain, starting with meters. McLaughlin said that the utility servers are Windows systems, but better defended and more isolated (they communicate only to gateway servers, which collect data from smart meters). Ulfar Erlingsson (Google) suggested looking again at software-based enforcement policies: "You may be assuming some hardware support is needed, but it is not." McLaughlin repeated the need for a supervisor mode for an inline reference monitor. Erlingsson replied that software-based techniques can be used to protect the reference mode, so they could use software fault isolation.

NETWORKS AND COMMUNICATIONS

Summarized by Femi Olumofin (fgolumof@cs.uwaterloo.ca)

- ***Evading Cellular Data Monitoring with Human Movement Networks***
  Adam J. Aviv, Micah Sherr, Matt Blaze, and Jonathan M. Smith, University of Pennsylvania

Adam Aviv began by describing HumaNet (Human-to-Human Mobile Ad Hoc Network), which is a network of humans and smartphones for providing unmonitored, completely decentralized, and out-of-band communication. Unlike cellular networks, which are centrally administered and prone to monitoring and censorship attacks, HumaNet avoids centralized controls by routing messages over mobile phones, at a cost of added delays to message delivery.

The design of HumaNet assumes that the movement patterns of mobile users are regular. It leverages the return-to-home principle, which assumes that a person is likely to return to places they visited in the past. The three main concepts behind the HumaNet protocol idea are that messages are not being duplicated as they travel though the network, messages are addressed to the recipient's likely future locations, and all local routing decisions are based on the movement history of the current carrier of the message. Message routes are refined with a local timeout and a global timeout to ensure that a message does not stay too long in the network. There might be some minor flooding when a message is close to the intended recipient (last mile flooding). The protocol makes local routing decisions by considering the profile of the mobile user's movement histories and ensures the sender's anonymity. They construct a user's movement history by clustering the GPS coordinates of geographical points she has frequented in the past.

They performed an evaluation of HumaNet using trace-driven simulation on a 20-day cabspotting dataset of 536 cabs in San Francisco. In comparison to similar routing protocols, such as epidemic flooding and probabilistic flooding, HumaNet requires a fixed number of messages for successful delivery, the same as for the random walk routing protocol. In terms of message latency, 76% of all messages are delivered within one day. In terms of successful delivery, 85% of messages are delivered for HumaNet, compared to the 76.3%, 60.3%, and 28.7%, respectively, for epidemic, probabilistic epidemic, and random walk.

Aviv also identified some challenges to overcome. First, the HumaNet protocol provides best-effort routing, which raised the question of how much reliability is needed for successful message delivery. Second, HumaNet routing is subject to the same set of attacks for peer-to-peer systems. Third, HumaNet requires periodic broadcast of a mobile phone's location information. Fourth, there arises the question of whether HumaNet can simultaneously provide both sender and receiver anonymity resistant to surveillance from the cellular service. They identified a k-anonymity scheme that resists Sybil attacks as a possible solution for preventing eavesdropping attacks on messages within the network. Some of the discussion questions included the feasibility of, and the number of resources required for, a successful attack against HumaNet, and what would need to be accomplished to motivate people to participate in HumaNet.

Prateek Mittal (University of Illinois at Urbana-Champaign) expressed concern that HumaNet might be weak against interception and routing attacks on anonymity. For example, a malicious user might go around town collecting people's location information and trying to map messages to sender or hijack messages meant for specific receivers. Rik Farrow commented on the need for sender's deniability; otherwise no one will use the system. Revealing a message and the intended destination is not sufficient for the users to be able to deny that they originated a particular message. Some form of encryption might help. Aniket Kate (University of Waterloo) raised some concerns with message secrecy and DoS attack vulnerability. Another attendee noted that clustering based on the mobile user's location is not enough; there needs to be some element of timing in the clustering process. For example, it might not be okay to route messages to the home of working people during the day, because they are likely to be at work.

- ***Challenges in Access Right Assignment for Secure Home Networks***
  Tiffany Hyun-Jin Kim, Lujo Bauer, James Newsome, and Adrian Perrig, Carnegie Mellon University; Jesse Walker, Intel Research

Tiffany Hyun-Jin Kim began this talk by outlining a vision of future smart homes, enabled by a number of technology trends such as user interfaces (UIs) for "everything," network communication, digital media, smartphones, smart meters and grids, and wireless medical devices. One central security and privacy challenge in smart homes is access control management for non-expert homeowners. Poor access control management could result not only in a privacy breach for an individual or family but in direct physical harm as well.

Kim subsequently discussed some of the challenges that make smart home access control management a unique and particularly difficult task. These include diversity of visitors, complexity and diversity of devices and resources, low sophistication of administrators, and social context in which a user might not want to reveal distrust for a visitor, but the user's distrust will become visible though the home access control policy (distrust revelation problem). Kim noted that some of these challenges might have appeared in some other contexts; however, a smart home environment presents a unique combination of these challenges.

Kim described a user study that forms the basis of their preliminary policy assignment. The study interviewed 20 people (8 males and 12 females). Participants' ages ranged from 20 to 60. The interview instructions asked participants to first list eight people who visit their homes on a semi-regular basis or who are potential future home visitors. The

participants were then asked to imagine what electronics and appliances would likely be in their future homes, and to define access policy for the identified devices.

They made three observations from the interview user study. First, they were able to validate some of the challenges anticipated for the smart home's access control management, such as the users being non-expert administrators; the complexity of home environments in terms of the number, diversity, interface support, and data stored on each device; the diversity of visitors; and concerns about distrust revelation. Second, they found three types of access policies (different from the current two-dimensional allow-or-deny policies), which are sufficient for defining desired policies: presence, logging, and asking for permission. The presence policy only grants access when the user is inside the home; the logging policy maintains detailed audit logs; and the asking-for-permission policy contacts the owner when a visitor attempts to use a resource. The three policies were used to derive two others: a combination of two or of all three of the policies (hybrid policies) and the always-deny policy. Third, they found four fixed groups of access-control rights to visitors, based on the duration of relationship and level of trust. These groups are: full control (grants complete control and full access to all devices for owners, close relatives, and household members); restricted control (grants full access to resources excluding entertainment and security systems for teenagers in the family); partial control (grants full access to sharable devices, such as a home telephone, for trusted friends); and minimal control (grants restrictive access to some devices for casual visitors).

Kim highlighted two areas of further research. The first is to conduct a full evaluation of the access policies and rights with a larger set of participants. The second is to work on the identified open problems of access control management for smart homes, such as dealing with multiple administrators in the home.

Hugo Straumann (Swisscom) identified the inconvenience of an unsophisticated smart home user always having to authenticate to a smart home device before changing system settings. Nathan Taylor from the University of British Columbia emphasized the place of an emergency override during a catastrophe. For example, in the event of fire, the babysitter might not know the code to open the front door. A related issue is how to activate the emergency override. Kim commented that the smoke detector coming on could be a way of telling when the emergency override should become active. Adam Drew (Qualcomm) commented on the importance of keeping things simple. One of the last things a working homeowner would like to do is to fiddle with home access control systems after dealing with access control at work. Since the device sits in your home, why can you not simply trust it? Kim said that the interview reveals that people restrict their definition of access-control rights to fixed groups of four, which is quite manageable.

- ***Scalable Anonymous Communication with Provable Security***
  Prateek Mittal and Nikita Borisov, University of Illinois at Urbana-Champaign; Carmela Troncoso and Alfredo Rial, ESAT/COSIC, IBBT-K.U.Leuven

Prateek Mittal began by identifying the requirement for Tor clients to maintain a global view of the network before they can construct circuits for anonymous communication as one of the main problems hindering the scalability of the network. An approach that requires clients to have a partial view of the network is desirable. A number of peer-to-peer approaches have been proposed, including Morphmix, ShadowWalker, Salsa, AP3, NISAN, and Torsk. However, all of these approaches are complex, require structured topologies, and are prone to attacks because they only provide heuristic security.

Mittal proposed two alternative solutions to this problem. The first solution is a peer-to-peer scheme based on reciprocal neighbor policy where the appearance of a peer node in the fingertables of other peer nodes is reciprocal. With this policy (also known as tit-for-tat policy), if a malicious peer A de-lists an honest peer B from its fingertable, then the honest peer B also de-lists the malicious peer A from its fingertable. A client constructs a route for anonymous communication using a random walk. The client first establishes a circuit with one of its random neighbors X. Next, the client queries X's fingertable for one of X's neighbors Y and then extends the circuit to Y, through X. This process is repeated to construct a circuit of any length. They also proved that this policy allows for better random sampling of Tor's node and substantially reduces the probability of route capture attacks. They also proposed some mechanisms for securing this scheme for both structured and unstructured topologies.

Their second solution is a client-server architecture called PIR-Tor. PIR-Tor leverages private information retrieval (PIR) to overcome the need for clients to know the IP addresses of all available Tor relays. In this architecture, such addresses will only need to be stored on some of Tor's central servers (e.g., directory servers). A Tor client intending to establish a circuit would need to query a few of these Tor central servers a fixed number of times to retrieve relays. Currently, the default number of relays needed to establish a Tor circuit is three. Using PIR minimizes the bandwidth needed to privately retrieve relays and prevents malicious central servers from knowing which particular set of relays the user has chosen for circuit construction. They also described how they overcame some of Tor's restrictions with respect to choosing relays. For example, Tor requires the first relay, called the guard node, to be a stable relay with a proven record of availability. In addition, this relay should be fixed for a particular client. Since PIR provides an effective means to trade off bandwidth for computation, they proved that the computation is still practical on modern commodity

hardware. They argued that deploying PIR-Tor will enhance Tor's scalability by an order of magnitude.

Someone raised some concerns about scalability when central servers have to sign and re-sign relay information after every change in the Tor network. Mittal commented that Tor has the notion of directory servers/authorities providing blind signatures on relay information and that the Tor network carries some overhead by having clients update their global view of the system every 30 minutes. Another person raised a concern that PIR-Tor does introduce some restrictions on client circuit construction. His reasoning was that the global view of the Tor network that users normally have is now being outsourced to some central servers. Someone responded by pointing out that most users would actually prefer PIR-Tor, since they will require less bandwidth to download the relays they need to construct their circuit. Besides, the subset of users who prefer to have a global view of the system can still download the entire database of relays from the central servers. In addition, the central servers may be required to send some metadata (e.g., exit policy) on available relays to the user before the user sends any query.

CATCHING MALWARE

Summarized by Quan Jia (qjia@gmu.edu)

- ***Retroactive Detection of Malware with Applications to Mobile Platforms***
  Markus Jakobsson and Karl-Anders Johansson, FatSkunk Inc

Markus Jakobsson opened by showing a market forecast for smartphones. He argued that smartphones' surging popularity has resulted in an accelerated rise in the incidence of mobile malware. Meanwhile, newly emerged mobile malware is becoming faster, stealthier, and smarter. However, the constraint on the battery power of smartphones is preventing the use of sophisticated antivirus software. Thus, approaches different from traditional malware detection methodologies should be adopted to ensure security.

From the consumer's point of view, usability and convenience are always primary concerns. In the case of security incidents, Jakobsson suggested that the ability to revert to a previous healthy state by clicking on the "Undo" button is often desired. This goal inspired the design of their retroactive malware detection mechanism. Before presenting the technical details of their solution, Jakobsson provided three key principles for malware detection: malware must be active to block detection; malware needs to stay in RAM to be active; malware is faster than flash and radio.

Under these guidelines, he described the main steps of the proposed malware detection process. First, all programs are swapped out from RAM, while malware may refuse to swap, so that it can remain active. Then, the "free" RAM will be overwritten by pseudorandom content generated by an external verifier. Similarly, active malware will again refuse to be replaced. At last, the keyed digest of all RAM will be computed and compared at the external verifier. In addition, the verifier times each step of this process. If an abnormal timing variance occurs at any phase or a digest mismatch arises in the end, a malware alert will be triggered.

Jakobsson emphasized that detecting latency is essential to defeat malware's attempts to fool the external verifier. As far as performance is concerned, experimental results produced by a prototype system showed the ability to finish each detection process within three seconds.

Paul van Oorschot (Carleton University) asked which device decides the correctness of the digest generated and what would be the follow-up action in case of incorrect response. Jakobsson replied that the external verifier always makes the decision. When digest conflict occurs, the entire RAM would be flushed before all programs are swapped back. This cleans up the active malware. Adam Drew (Qualcomm) asked how kernel-affecting malware, for example a rootkit, could be detected. Jakobsson responded that the entire operating system is swapped out during the detection process so that a rootkit can also be exposed. Angelos Stavrou (George Mason University) asked whether event-driven malware could bypass such detection. Jakobsson said that malware of this kind makes no difference, in that it needs to be active in RAM to listen for its trigger. Finally, someone asked what measures are employed to counter phone cloning. The SIM card of each phone is used to mark its unique identification.

- ***Scalable Web Object Inspection and Malfease Collection***
  Charalampos Andrianakis, Paul Seymer, and Angelos Stavrou, Center for Secure Information Systems, George Mason University

At the very beginning of his talk, Angelos Stavrou indicated that the goal of this work is to collect URLs where malware originates. This goal is achieved by constructing a honeynet with the proposed framework that does automatic malware analysis. To build a system for this purpose, full virtualization techniques—for example, VMware ESX and Xen—are inefficient in that they are expensive and thus not scalable. Therefore, Stavrou and his team opted to design their architecture using WINE combined with lightweight virtualization. To further describe their framework, Stavrou explained that they used OpenVZ for building isolated containers. Each container is installed with a Debian Linux operating system and a modified version of WINE. An unpatched instance of IE running within a container is responsible for visiting supplied URLs and executing downloaded objects. The customized WINE installation has a built-in memory allocator that is able to detect NOP sleds. By this means, the URLs that are spreading heap-spray exploits will be identified and logged.

Stavrou then presented an experimental evaluation of their system, showing that heap-spray-based exploits can be successfully detected as expected. What's worth mentioning is that the system not only could identify known exploits but also is able to catch many zero-day exploits. Meanwhile, im-

pressive data was shown to prove the superior scalability of lightweight to full virtualization. As for the limitation of this work, Stavrou said that the current framework could only enforce heap-spray memory detection. Other exploit detection mechanisms need to be integrated with the system in the future so as to enrich its functionality.

Carlton Davis (École Polytechnique de Montréal) asked whether the framework could detect malware carried by file droppers. Stavrou answered yes and reiterated the premise that the malware should be heap-spray based. Someone asked why IE was chosen. IE is the most popular target for attacks. Were static IP addresses used for the clients, and how did the malware server react? They had the resource of an entire C class IP pool and used dynamic IP addresses for each client. This protected their clients from being remembered by a malware server. Wietse Venema (IBM Research) asked if different OSes were used to run each individual exploit. Starvou responded that WINE in different containers was configured to mimic different versions of Windows. Because of this, they were able to observe some malware adjusting their behavior to adapt to such change. The last question was about the source of new malware URLs. Stavrou said they used Google's safe URL on the one hand and extracted URLs from GMU network users on the other.

1st USENIX Workshop on Health Security and Privacy (HealthSec '10)

August 10, 2010
Washington, DC

INVITED PANEL: MEDICAL DEVICE SECURITY AND PRIVACY

Summarized by Leila Zucker (leila@motherzucker.com)

- ***Ten Years of Insulin Pump Therapy: From User to Researcher***
  Nathanael Paul, Research Scientist, Oak Ridge National Laboratory

Nate Paul told us that he received his first insulin pump in 2000. He gave a brief overview of diabetes, how the insulin pump works, and how the systems may be vulnerable. The pumps can be very complicated, and there are classes to teach you how to use them. Newer pumps have an increasing number of features, including remote wireless programming and the ability to update settings by personal computer. While these features improve effectiveness, they also represent the threat of exploitable vulnerability and decreased safety. Approximately 13 different attacks have been described to the FDA. In looking for solutions we must address both issues.

Session chair Kevin Fu asked each of the presenters to describe the biggest research problems for security and privacy. Paul answered, data transmission. Don't get attached to a specific device, but focus on the entire system. Fu then asked about incentive systems for improving security when there is shared responsibility. Paul responded that manufacturers are aware of compliance, safety, and security. Revealing source code would be a good step, or the FDA could review source code. Carl Gunter (University of Illinois) asked whether the 13 problems with the insulin pump could be solved by best practices or whether they required a novel approach. Paul felt that general solutions were needed that would apply to all devices, both implanted (e.g., pacemaker) and partially embedded (e.g., insulin pump).

- ***FDA Regulatory Perspectives on Cybersecurity***
  John F. Murray Jr., Software Compliance Expert, United States Food and Drug Administration, CDRH/Office of Compliance

John Murray said that confusion seems to exist about what the law requires vendors to do. The FDA rules only apply to manufacturers, not to software vendors or clinical facilities. Manufacturers must validate patches. Viruses have caused major disruptions to clinical information systems, but there is no formal reporting of cybersecurity issues. Vendors have reportedly told hospital IT staff that they can't install security patches "because of FDA rules." Therefore we need FDA outreach to the clinical IT community.

The law requires that deaths be reported to the FDA and the manufacturer, serious injury to the manufacturer only, and potential injury or death to MedWatch on a voluntary basis. The manufacturer must report if there is any chance a device may cause a death or any indication of quality deficiency (go to http://www.fda.gov and search for "cybersecurity"). The FDA addresses safety, not security, concerns. To solve the problem of medical device security will require the efforts of IT infrastructure vendors, healthcare IT administrators, and medical device manufacturers.

Paul Jones at the FDA is doing research on device tracking, secure record transfer, and the question of whether to allow patients to take records home. The current focus is on functionality, but security and safety issues need more attention. The IAC standards organization is addressing the issue of different stakeholders negotiating safety and security, and voluntary standards will be published soon. However, the FDA will be highly dependent on the cooperation of device manufacturers. The FDA's inability to review every line of code supports the idea of having medical device software all be open source. Please feel free to contact Murray with any questions (see http://www.fda.gov/MedicalDevices/Safety/MedSunMedicalProductSafetyNetwork/ucm127922.htm).

- ***Killed by Code: Software Transparency in Implantable Medical Devices***
  Karen Sandler, General Counsel of the Software Freedom Law Center

Karen Sandler told us that two years ago she got a pacemaker/AICD (automated implantable cardioverter-defibrillator). She was very concerned about the safety of the software in the device, particularly when she found out that she could not obtain the code to check it herself. She finally settled

for an older device with no wireless component She is now researching pacemaker software, which on average has one defect per 100 lines; 98% of software failures could be detected by all-pairs testing. Security through obscurity just doesn't work. However, with free and open source code, users have the ability to independently assess the system and risks, patch bugs easily and quickly, and remove dependence on a single party. Shared software does not mean unprotected devices—you can still use encryption.

The FDA does not review source code, only manufacturer reports. There is no clear set of mandatory requirements for software and no repository of source code, which prevents patients from suing under state product liability laws. All software should be made safe: medical devices, cars, voting machines, financial markets. See www.softwarefreedom.org.

In the discussion, Sandler emphasized that with wide adoption of implantable medical devices, the biggest research problems for security and privacy concern the need for open-ended transparent solutions so that security can be verified. Since the open source world is about collaboration, with shared systems it is more likely that everyone will understand them. Avi Rubin (Johns Hopkins) interjected that the many-eyeballs theory works in Linux, but in the real world a hacker can find vulnerabilities, so patches might not come out quickly enough. Sandler replied that it's been seen that not publishing does not stop attacks. Umesh Shankar (Google) felt that it's really about transparency. While there are not many hackers, plenty of people want to test for vulnerabilities, but there is a question of manufacturer liability. Sandler said that she can think of hacks for her pacer, and primarily wants transparency.

SENSORS, CLIENT DEVICES, AND MOBILE HEALTH

Summarized by Leila Zucker (leila@motherzucker.com)

- ***Protecting E-healthcare Client Devices against Malware and Physical Theft***
  Daisuke Mashima, Abhinav Srivastava, Jonathon Giffin, and Mustaque Ahamad, Georgia Institute of Technology

Daisuke Mashima said that in their setup, data is stored in online repositories and a threshold keys system is used to control access. The client device includes one VM for the user interface and another that holds one key and handles communication. The other key resides at a logging service, although a human administrator can also provide a key. Mashima went over ways to handle issues if a client device is compromised and for eliminating single point of attack, including the threshold signature scheme, human authority, online monitoring system, user virtual machine, and firewall.

- ***Can I access your data? Privacy Management in mHealth***
  Aarathi Prasad, Dartmouth College; David Kotz, Dartmouth College and ISTS

Aarathi Prasad said that if you want patients to use EHR, you need to instill confidence in them. For example, a patient sharing jogging data from her mobile phone with a wellness advisor might not want the advisor to see her jogging route. What data do we need, then, and when do we collect it? A patient may remove sensors and forget to reattach them. Another issue is that many doctors believe patients cannot tell what data to share. Several items were mentioned for consideration: Do we retain old data? Should backups be retained, with or without the patient's knowledge? Query and response should be fixed format. There should be user interface requirements that are unambiguous and use few medical terms. Finally, future work includes learning patient privacy concerns, identifying benefits and trade-offs, and determining what to delegate to doctors.

Attendee Vince (last name and affiliation not stated) inquired about what to do if a patient revokes access. Prasad replied that they are researching this now. The session chair, Tadayoshi Kohno, asked about patient privacy. Prasad replied that while doctors need access, patient privacy is an important consideration. Kohno then asked about emergency situations, people who can't afford a phone, and other countries. Prasad responded that researchers can take phones to rural areas to collect information. An attendee noted that most people don't have a single point of access to the healthcare system or even always know their doctor's name, and privacy laws vary by country. Prasad added that some cultures don't have the concept of privacy, so how much of their data can you use? Kohno said that consent management is important; doctors should take only information needed for care. He then asked if there is research on how much patients want to be involved with management of their EHR. Prasad said they are working on that.

- ***Using Trusted Sensors to Monitor Patients' Habits***
  Alec Wolman, Stefan Saroiu, and Victor Bahl, Microsoft Research

Alec Wolman addressed the problem that patients often do not follow doctors' instructions. It can be difficult to manage chronic diseases such as high blood pressure and diabetes outside the office. Smartphones can be used with sensors to change the status quo by assisting patients in monitoring their habits: an accelerometer can be used for exercise, a camera for diet, a pressure sensor for their pillbox, body sensors for heart rate and blood pressure, and so on. Financial incentives can also be used to change patient behavior. But data gathering must be done in a trustworthy manner. Trusted sensors include laptops with TPM chips or smartphones with ARM's TrustZone. We can use trusted computing primitives to preserve the integrity of sensor readings with digital signatures and verification. He outlined two approaches: software only (trusted VM) with no barrier to deployment but also with no wireless security, and simple hardware changes such as tamper-resistant casing. Trusted sensors must protect against malicious use and ensure that users do not fabricate readings.

An attendee wondered who would benefit financially from attacking. Generating false claims is a bigger risk than monitoring sensor data. Wolman said thatraw data does have significant financial impact. Hackers want raw data for

making fraudulent claims. In response to the chair asking about the future of medical sensors, Wolman said that monitoring eating disorders with weights recorded by sensors would be useful, as patients often report false weights. Carl Gunter (University of Illinois) observed that his cell phone surreptitiously spying on him and reporting to his doctors would not be his idea of a killer app. Wolman said the user could be in control of what readings are taken and what is revealed to whom. How likely are these apps? Mobile devices can take pictures of checks to be filed with your bank. Security is a big challenge; so are energy management and battery life. The Chair asked how one could monitor a patient who cheats. Wolman replied that there is no way to stop this currently unless you use multiple sensors.

Nate Paul (Oakridge National Labs) pointed out that smartphones have been used with insulin pumps, but that means carrying an additional device. Wide-scale attacks on insulin pumps could have some financial advantage. Jack Lacy of Intertrust asked about data integrity vs. privacy and a patient needing selective control over sharing data. Perhaps offer incentives: if you don't opt in, you are penalized by insurance companies. Chase replied that it's a question not only of who gets access, but also of what they do with the data. The chair next commented that some EHRs allow a designation to not reveal certain data, but it might be revealed in a free text note. Wolman said that it's critical to put the patient in control. Is there a way to penalize the patient if they do not comply? We need to be mindful of this when creating incentives. Gary Olson (Intertrust) asked how much trust is enough. Do you need hardware, or is software sufficient? Cost is a problem. Wolman replied that hardware is coming, independent of medical apps. As for trust, you want to protect yourself not only from users but from malware. The final question by the Chair was, Is runtime integrity enough? ARM is more flexible than TPM with runtime integrity.

POLICY FOR HEALTH RECORDS

Summarized by Joseph Ayo Akinyele (jakinye3@jhu.edu)

- ***Practical Health Information Exchange using a Personally Controlled Health Record***
  Ben Adida, Isaac S. Kohane, and Kenneth D. Mandl, Children's Hospital Boston and Harvard Medical School

Ben Adida said that PCHRs represent a paradigm shift, with medical records controlled by the patient as opposed to the Nationwide Health Information Network (NHIN) model, which allows access to records via a Web portal. In addition, patients visiting different specialists for various purposes means that records must be aggregated in one location (the PCHR). With PCHRs, data can be aggregated from a variety of sources, including from implantable medical devices such as pacemakers and defibrillators. Patients can annotate their records and share selectively with their physicians. In this model, patients determine, through the PCHR, who gets access to their data, and clinics or hospitals connect to the PCHR to access the patient's records; in the current, provider-centric NHIN model, the focus is on provider-to-provider data sharing, and patients have to independently give each provider access to their records.

Adida argued that Health Information Exchange (HIE) can be mediated by the PCHR and that once data is shared with the physicians, patients are mainly concerned with who else may have access. But if patients are given tools to annotate, update, and share their data, this could create a very powerful health record system. He concluded with comments on the future of the PCHR concept, involving using email addresses to locate health records and PCHR format standardization across healthcare providers.

Avi Rubin (Johns Hopkins) commented that despite the common belief that patients should have control over their records, most doctors mistrust records received from patients. Adida replied that the idea of patients having control has evolved because healthcare is fragmented today. Hospitals rely on out-of-band mechanisms to share records with other hospitals. With PCHRs, patients can now take their records from one doctor to the next. However, safeguards must be in place so that patients do not unwittingly share data. One audience member asked how much thought has been put into the ecosystem of patients controlling their records. For example, in the Indivo PHR system, when considering records for children, parents have more access to their children's PHRs than to their own. Further, any sensitive data is not included in the PHR, because of the difficulty in managing that data.

- ***Technology Companies Are Best Positioned to Offer Health Record Trusts***
  Shirley Gaw and Umesh Shankar, Google

Umesh Shankar discussed the notion of a health record trust as an independent archive of patients' medical data in which patients ultimately have control over how their information is released. These trusts guarantee that data from any time period can be retrievable without loss of information. For instance, a patient with a chronic disease may want to see the progression of her disease over time, but if the practice or clinic goes out of business, the data could be difficult, if not impossible, to retrieve.

A service that can meet demands for high availability, data integrity, and provenance is achieved best by technology companies. Technology companies are more diverse and do a better job than government-led IT in handling large-scale projects. Most large-scale IT projects in fact do and should fail, but the government wastes millions of dollars on projects that end up not working. If ten tech companies compete on an IT project, perhaps three provide a good solution, making the odds of a working solution much better than solely with the government. EMR vendors have a lot of experience managing medical data, but usually only for a single hospital or HMO. The biggest challenge is the aggregation of data from different sources, and EMR vendors are not built to support such integration with other vendors.

Tech companies are prepared to work on a problem of providing record trusts on a large scale that requires high availability, integrity, and redundancy. Shankar pointed out that people expect Google services to run all the time, and when those services are unavailable, Google is embarrassed. This is how it should be for health records. However, tech companies cannot solve this problem alone. They need government and EMR vendor collaboration to establish public-key infrastructures for trusts and to define interoperability standards between institutions. These are some of the issues that tech companies should not solve in an ad hoc fashion.

One audience member asked if patients are to trust tech companies with their data and whether the incentives are aligned with the patient's privacy. What is the business model for Google Health? Shankar replied that Google does not make money from providing health services to patients or from securing their data. Although Google Health does not conform to HIPAA regulations, Google's privacy guidelines have the patient's interests in mind. Another audience member asked whether placing all the trust in a tech company creates a failure and availability risk. Why not consider an open federated model for managing record trusts? Shankar agreed that the records should be fetched from different providers, but argued that the records must still reside in a central repository.

- ***Policy Management for E-Health Records***
  Maritza Johnson and Steven M. Bellovin, Columbia University

Johnson began by explaining that EHRs are records created and maintained by institutions such as hospitals, and patients may or may not have access to the information. This notion of health records is different from PCHRs, which are maintained by patients. Existing access-control EHR systems allow access to all patient health records, based on successful authentication to the EHR system. User access (including by nurses, doctors, etc.) is audited by the system, and patients must monitor their own records for unauthorized accesses. An exception to this rule is the EHRs of celebrities, professional athletes, and chemotherapy patients admitted at the hospital. Because these types of patients are high-profile, strict access controls are enforced.

New mechanisms are needed to support and control EHR-sharing between hospitals. Currently, ad hoc out-of-band mechanisms such as email, fax, or mail are used to share EHRs. Johnson argued that an adequate architecture that supports access policies must be developed, and she questioned who will manage the access policies and with what mechanism. So far, the focus has been on the adoption of electronic records, not how they will be shared, how access is controlled, or even what those access policies will be.

In the current literature there are two kinds of approaches to EHRs.Human-centered approaches focus on the interactions doctors have with EHRs over paper-based charts in day-to-day activities. Computer scientists focus on the architecture for EHRs to support sharing EHRs. Johnson discussed two possible types of access control: preventive and audit-based. Preventive (similar to file-based) access determines access policies a priori; audit-based relaxes preventive policies for emergency situations. Finally, usable policy tools are needed to handle the difficult task of creating and managing fine-grained access policies for EHRs.

- ***Dr. Jekyll or Mr. Hyde: Information Security in the Ecosystem of Healthcare***
  Joseph Cooley and Sean W. Smith, Dartmouth College

Sara Sinclair, speaking for Joseph Cooley, said that policymakers mandate that medical data should be protected in a particular way, but deployed mechanisms do not match the policy or align with daily practice. Once these mechanisms are put in place, clinicians work around the mechanisms when they prevent them from getting their work done.

To achieve usable security from a healthcare provider and patient perspective, the authors propose retrieving user feedback. Acquiring feedback is a proven approach to understanding issues between a system and its users. This same approach is proposed to help improve the ecosystem of the healthcare environment. The authors argue for a practical approach which includes spending time with users, performing observations with users and a system, and retrieving system logs to elicit feedback. The authors' goal is to equip clinicians, policymakers, and developers with information to be able to implement such mechanisms.

The authors argue that to be worthwhile the feedback process should be easy and painless. Users should not suffer negative repercussions for providing feedback. For instance, if a clinician shares a password in order to get her job done, then she may be subject to certain penalties by the hospital. In addition, the process should reward the users such that they are motivated to help improve and build trust in the system. If closed loop feedback is provided and it is possible for users to inform the system, users will have greater trust in the system.

Avi Rubin asked whether the role for technologists is to develop solutions and leave decisions to policymakers or to develop solutions that influence decisions one way or another. It is impossible to design a one-size-fits-all system that satisfies technology and policy requirements. However, technologists should not blindly design systems based on preconceived needs of a system. Technologists need to collaborate with users, usability experts, social scientists, and clinicians to build a system that satisfies both requirements.

- ***Privacy Challenges in Patient-centric Health Information Systems***
  Anupam Datta, Carnegie Mellon University; Nipun Dave and John Mitchell, Stanford University; Helen Nissenbaum, New York University; Divya Sharma, Carnegie Mellon University

Helen Nissenbaum and Anupam Datta presented this workshop paper on the privacy challenges in personal health record (PHR) systems. Nissenbaum questioned what the rules or policies should be that govern the inflow and outflow of information in PHRs and how to formalize these rules and

enforce them in the PHR systems. PHR systems such as Google Health and Microsoft HealthVault provide aggregation of data from diverse sources, offer patient control, and allow for customization according to the patient's needs. For example, these systems can determine a patient's risk for diabetes simply by analyzing their PHR. With PHR systems, healthcare providers can now extract all sorts of interesting data from anonymized PHRs such as for advertising or for public health purposes.

Nissenbaum argued that the notion that patients have full control over their PHR contradicts and is incompatible with the idea of practitioners using the patient's PHR as a basis for medical care. Doctors and clinicians are concerned with the integrity of patient data and usually prefer that the patient data come from their colleagues. Nissenbaum asked what model patient health records should follow: a patient portfolio model (i.e., patient controlled), a credit-report model (i.e., institutionally managed records), or a trust-based model (i.e., third-party managed records). Because each model offers different levels of patient control, the model selected must promote the values and purposes in the context of providing medical care.

Datta then discussed the challenges of representing policies and enforcing those policies using traditional access control mechanisms. He referenced their previous research that analyzed the HIPAA requirements and created a system to formalize those requirements in logic. He argued that such requirements cannot be enforced using traditional access control, due to interpretations of the HIPAA rules. For example, the HIPAA rules use terms such as "belief" or "trust" that are subject to various interpretations. The authors proposed a hybrid approach which incorporates proactive access control and auditing to enforce the HIPAA rules on PHRs.

An audience member asked whether data integrity based on the source of the information is considered in PHR models. For example, a doctor could add an incorrect diagnosis into a patient's records, for a variety of reasons. Nissenbaum replied by referring to the different models of health records she discussed during her talk that offer options for patients. The issue of data integrity of records is controversial, as the model chosen for the health records will dictate how information is controlled.

DEVICES

Summarized by Aarathi Prasad (aarathi@cs.dartmouth.edu)

- ***Security That Is Meant to Be Skin Deep: Using Ultraviolet Micropigmentation to Store Emergency-Access Keys for Implantable Medical Devices***
  Stuart Schechter, Microsoft Research

Pacemakers and implantable cardiac defibrillators are increasingly becoming wireless. It has been shown that unauthenticated commands can change the device state. An obvious solution is to add authentication and check if the commands are from an authorized party. But how do we distribute this key? This issue can be solved if the key could be something you know, you have, or you are, where the "you" implies the emergency health care provider or the patient. These can also be interpreted as what you forgot, what you lost, or what you used to be! Something that you know can't be given to all HPs, especially if the patient is unconscious. Something you have could be an object that you possess, but if you lose it or forget to carry it with you, who gets access? An alternative to this is authentication by proximity. Something you are implies that the key could be a biometric; but this might be difficult, especially if the device is implanted. Medical tattoos can be placed next to the scar that marks where the device has been implanted. It should be in human readable form, in case the tattoo reader fails. This key can be generated by the patient's device. The tattoos could be ultraviolet for privacy, since other tattoos might be visible. When their clothes are off, patients can hide the tattoos using sunscreen.

- ***Privacy Challenges for Wireless Medical Devices***
  Brent Lagesse, Oak Ridge National Laboratory

Device usage can leak information such as conditions, patient actions, or device type or model. Adversaries can determine what the patient is doing by profiling using communication patterns. So if an adversary knows the protocol used, he can determine who among a crowd has a particular device implanted. Adversaries can launch spoofing, replay, and denial of service attacks or, knowing that a patient has a certain condition, physical attacks. A patient could also be subject to discrimination by an employer or insurance company. Several approaches have been taken to protect privacy—encrypt the data (traffic patterns might still leak information), mask communication so that you will be in the set of possible sets of insulin pumps (k-anonymity), and mixes (pass the information to all other devices that the patient is carrying such that it is possible to hide what devices are being used).

New approaches are being studied to reduce attacks, remote to physical, to provide practical privacy (by changing cyber to physical attacks) and to prevent wide-range scanning. Keep in mind that as a researcher, you need to take a minimalistic approach—sensors don't have to be general-purpose computers. The device should use common protocols; if only one device is using a strange protocol, it is easy to identify.

- ***Insulin Pump System Security***
  Nathanael Paul, CSIIR, Oak Ridge National Lab; David C. Klonoff, Diabetes Research Institute, Mills-Peninsula Health Services

Medical devices can communicate with patients, caregivers, etc. There are different components to this system that interact with the device and with each other. An example of a remote-control device is a glucometer which tells patient

about blood glucose level. This glucometer could be recording continuously. A smartphone can be used to control a glucometer remotely and can also act as a data recording tool. As a monitoring tool, the smartphone can calculate how much insulin the patient needs, store this data, and use it to record data from the patient. As a controlling tool, it can issue several commands.

Physicians are increasingly using mobile devices to control or monitor patient condition. But worms can spread from phone to phone through Bluetooth and we need to determine how this will affect patients.

- ***Is Bluetooth the Right Technology for mHealth?***
  Shrirang Mare, Dartmouth College; David Kotz, Dartmouth College and ISTS

A communication technology for medical devices should be: (1) secure—it should authenticate transactions and ensure that the data is correct; (2) private—encryption does not provide enough privacy protection for patients; (3) reliable—the device should be able to resist interference; (4) scalable. The *E*0 cipher was used to provide security, though it was not secure enough. JW was used for those devices without I/O. Privacy became an issue, since headers of the data packets contained MAC information, which could be used to identify devices and link all data transactions. Reliability was achieved through Bluetooth's hopping pattern and channels were reduced from 79 to 40. But how will Bluetooth piconet interfere with other Bluetooth technologies?

There are alternatives to Bluetooth. The Sly-Fi protocol encrypts headers and provides unlinkability. We could use the human body as a communication channel, which is more secure and private. Galvanic transfer involves attaching electrodes to the skin and sending electricity through the human body. A 2 mbps throughput is achieved. Another option is body-coupled communication, which achieves less throughput but is power efficient, but is this safe? There are other issues as well.

- ***On Usable Authentication for Wireless Body Area Networks***
  Cory Cornelius, Dartmouth College; David Kotz, Dartmouth College and ISTS

If we want systems to be used, we should make them usable, so that the patient shouldn't have to do anything. Body area networks are a bunch of devices that send data to a gateway. Data can either be sent to a cloud service or be stored on a mobile phone. Devices such as Fitbit and Nike plus are available in the market now. We need to provide usable and secure authentication so that doctors can be confident that data is coming from the actual patient. The problem being tackled is a weak version of this, where the cell phone or gateway is trying to determine if all sensors are on the same person. The strong version of this problem confirms that all sensors are on the actual patient. This authentication is necessary in a scenario where an elderly couple might accidentally swap sensors. A solution to this problem is wireless localization. Even though the body might attenuate signals, we can still detect if the sensors are within some bodily distance.

In the Q&A the authors were asked what prompted their research—had there been any such attacks? They answered that some attacks might not leave evidence behind, and, in order to protect their business, manufacturers might not reveal that attacks had occurred. If an attack does happen, it might be difficult to patch; hence it is better to prevent attacks. It is also important to anticipate attack scenarios, in order to identify the important security metrics. The switch from wired to wireless has exposed the devices to imminent attacks. But wireless does help treatments. We need to consider battery life too. Wireless technologies drain batteries, and unauthorized commands can launch a DoS attack on the device by draining the battery. You can't replace batteries on implantable devices easily. The patient can lie about the data, so that could be an attack too.

How do you evaluate proposals and next steps? With a user feedback form. Why are medical tattoos better than biometrics? Using biometrics, you are not authenticating the health care provider. Biometrics should not be changing and hard to read—for example, a heartbeat would change in a dying patient. The patient will recognize that his privacy is being violated when someone tries to access the tattoo, unlike a retina or fingerprint scan. If someone asks to check the area on the body near the scar, the patient will realize that the person has no need to do so.

SHARING DATA

Summarized by Tamara Denning (tdenning@cs.washington.edu)

- ***A Risk Management Framework for Health Care Data Anonymization***
  Tyrone Grandison, IBM Services Research; Murat Kantarcioglu, University of Texas at Dallas

The goal of this research is to share data sets for research in an anonymized way; the ideal result would be to anonymize data sets so that they are protected from all re-identification attacks. This work takes a slightly different approach from other research in this area by embracing a more practical approach: not 100% guaranteed privacy, but providing a user instead with information about the amount of risk that is left.

The risk management framework for health care data anonymization incorporates: (1) the chance of re-identification of sensitive data; (2) the repercussions of data reidentification; and (3) the utility of sharing the data set in question.

The researchers propose that the parameters of the risk model can be estimated using publicly available data. The risk management framework must also incorporate a way

to tune anonymization based upon risk estimates (and feedback).

- ***On Resolving the Privacy Debate in Deidentified Neuroimages***
  Nakeisha L. Schimke, Mary Kuehler, and John Hale, University of Tulsa

The topic of this research is finding a way to deidentify neuroimages such as CT, MRI, and PET scans. These scans are images with high spatial resolution that includes facial data. Personally identifying information (PII) can be stripped from the image's metadata, but there is a chance that a person may be identifiable based upon the facial data in the image.

One approach is to remove all content from the image except for the brain tissue; however, this can result in the loss of some brain tissue imagery, and there is no standard for this kind of deidentification process. As a result, neuroimages using different deidentification processes may not be comparable in research studies.

The researchers are currently experimenting with different reidentification techniques in order to study whether it is feasible to reidentify visually deidentified neuroimages. Based upon their findings, they may also investigate possible mitigation techniques.

- ***Securing Medical Research Data with a Rights Management System***
  Mohammad Jafari, Reihaneh Safavi-Naini, and Chad Saunders, University of Calgary; Nicholas Paul Sheppard, Queensland University of Technology

The motivation behind this research is to be able to share data for research while simultaneously respecting patients' privacy. Current approaches include anonymizing data sets by adding noise, which can result in losing relevant data, and using access control policies.

The researchers propose using DRM mechanisms in order to control access to medical records. Specifically, they are addressing "bench-to-bedside" medical research, where clinical information is repurposed for medical research. The authors suggest an approach where data is always encrypted and a trusted agent examines the data and a license in order to reveal decrypted data as allowed by the license.

The authors released a 2009 technical report describing a Sharepoint implementation of such a system where data is presented in DRM-protected Excel files. In future work they hope to refine access roles, handle data from multiple sources, and extend their system to operate in the cloud.

Questions for the panel of presenters included the nature of PII: specifically, where does PII end? If you bring in enough contextual information, almost anything can become PII. One workshop participant suggested that rules should be attached to pieces of medical information that define whether or not the information is PII given the context.

A question regarding the risk management framework was how the risk can be boiled down to a scalar number, when the risk of reidentification might be distributed across the members of a population. One proposed approach was to have adaptive fuzzing, where unique individuals are overgeneralized (fuzzed) and individuals closer to the norm are undergeneralized. This led to the question of how one can assign a uniqueness score to an individual based upon the fields of the medical record.

In terms of adding noise to data sets, concerns were expressed that the noise can affect downstream science. In addition, medical researchers in general dislike working with fuzzed medical records. To further complicate matters, the anonymized records do not generally contain metadata about the anonymization techniques used to add noise to the data.

More general concerns were expressed that the security community may not completely understand the domain-specific problems related to working with medical records. Additionally, a workshop participant suggested that patients and study participants do not have an accurate or complete understanding of what it means to have their records anonymized or of the various degrees of privacy different anonymization techniques can offer.

APPROACHES FOR HEALTH RECORDS

Summarized by Aarathi Prasad (aarathi@cs.dartmouth.edu)

- ***Beefing Up a Health-Data Ecosystem: Struggles and Successes from Microsoft HealthVault***
  Jim O'Leary, Microsoft Health Solutions Group and University of Washington

O'Leary discussed the common problems faced by the HealthVault team and how they were handled. First, he talked about authentication. Security provided by HealthVault is "weak" because the credentials used to log in to HealthVault are shared with "lesser integrity systems" such as email, calendar, and Xbox live accounts. HealthVault depends on third-party providers, such as LiveId and OpenId, that support authentication protocols. It includes dependencies in systems, but providing these options gives redundancy. Authorization is at two levels—a user wants to share his health information with another user or with an application. This presents another struggle, since users want granularity. They want to control what data is going where and to be able to track it throughout the system. But this produces usability issues.

Then O'Leary talked about the ecosystem security model. Microsoft has to depend on its partners. When faced with a blue screen, end users might blame Windows, but it might be due to some third-party error. The same issue happens in HealthVault when a partner loses data—the blame is on HealthVault. He briefly mentioned the shared data problem of multiple people sharing data from multiple platforms,

which introduces more trust relationships and attacks. Solutions to the attacks are presented as security tips in a public white paper.

- ***Using the Wave Protocol to Represent Individuals' Health Records***
  Shirley Gaw and Umesh Shankar, Google

Shankar said that attribution is important in health records. In the latest Google Health UI, a mouse-over a particular data point on a graph from a lab test will tell you where the point of data came from. How do you preserve attribution over time and in aggregated data? Dr. Dre sends his diagnosis to Shankar's PHR from his EMR, saying Shankar broke his ankle and tore his Achilles tendon. After two months, his ankle has healed. Shankar updates his record, adds an end date. Hence the real diagnosis done by Dr. Dre is gone due to the change done over time.

Or suppose there was an error, accidental or deliberate. The local copy is gone when you delete something. How do you update the central server? When the doctor tries to synchronize the data with Shankar's updates, how does the doctor know what to take from or send to Shankar's PHR? A "diff" will not help, since a deletion occurred. There should be a common notion of the state of a patient's records. This can allow for bi-directional updates. Also, a history should be maintained, not just the current state. Wave protocol can be used for this purpose (http://www.waveprotocol.org/).

- ***EBAM: Experience-Based Access Management for Healthcare***
  Carl Gunter, University of Illinois at Urbana-Champaign; David Liebovitz, Northwestern University; Bradley Malin, Vanderbilt University

Carl Gunter explained that identity and access management are crucial enterprise functions in health care organizations (HCOs), but insufficient attention is given to the process of access control. What is the fundamental problem? Accountability versus enforced control: HCOs give access to everyone, assuming they will use it properly. Professional ethics are set up by the government and this is too difficult to set up as enforced control. Access logs are raw and factual and should be converted into information that can be understood in a manner that we desire. EBAM takes into account what happens in the system and feeds it back in, so that over time the model would evolve into something close to ideal.

The enforced-control model generates the access logs, which can be compared to the ideal model. The access logs combined with the ideal model give you good knowledge of what you want in your organization, and though them the enforced control model can evolve. The EBAM approach involves generating models based on audit events and attributes. These are used to create workflows and group health providers into social networks. Rules and actions are developed after analyzing the results.

Experience-based systems have been used for a long time; successful ones include spam filters and intrusion detection. This technique can be used only in applications that tolerate false positives and negatives. There is a strong demand to catch violations in access control and there is a debate over what technologies should be used. There is also the challenge of health information exchanges between organizations.

- ***Fine-grained Sharing of Health Records using XSPA Profile for XACML—An Extended Abstract***
  A. Al-Faresi, B. Yu, K. Moidu, and A. Stavrou, George Mason University; D. Wijesekera, National Institute of Standards and Technology and George Mason University; A. Singhal, National Institute of Standards and Technology

How can we collect the different actors, patient's health information, and other parts of the record, such as psychological notes, into one model? asked Wijesekera. To what extent has XSPA captured all these scenarios? HITECH implies that the patient should have delegation rights. On the other hand, PHI can be disclosed without an individual's authorization for certain national priority purposes. Health records contain different views for the patient and the healthcare provider. When the healthcare provider needs access to some information, he sends the request to the patient, without knowing whether the data was actually derived from those records that he had access to. Some changes are required in the XSPA model in order to fulfill its central purpose.

- ***An Anonymous Health Care System***
  Melissa Chase and Kristin Lauter, Microsoft Research

Melissa Chase pointed out that privacy is a huge concern in healthcare, so we should be careful to reveal information only when necessary. Doctors, nurses, insurance companies, pharmacies, etc., need to see patients' health records, but not necessarily all the information that they contain. So a health record system should reveal as little as possible, while allowing the consumers to access the required information.

One technique is to use Anonymous Credentials or Minimal Disclosure Tokens, which ensures that the service cannot identify the user. An example scenario involves user Alice, who gets a policy token when she registers with an insurance company. Her doctor uses Alice's policy token for some transactions. When the doctor bills the insurance company, he uses an anonymized token for the procedure. The insurance company learns that some patient had some procedure done. Similarly, the doctor can send an anonymized token for a prescription to the pharmacy and a prescription token to both the pharmacy and the patient. The pharmacy can send an anonymized token for the prescription to the insurance company as well. This ensures that only required information is shared with others. The anonymous policy token needs to contain only the information that the recipient requires.

The limitations of the system are not technical. The take-home message is that we should be thinking about what information should be revealed and reveal only what is necessary.

Many issues were brought up in the ensuing discussion. There could be other factors, not just bias, that could become barriers to adopting the techniques. For example, insurance companies would want patients to buy medicines from their authorized pharmacies. Could the authentication methods in HealthVault be used by real patients in real-world circumstances? HealthVault targets all kind of people. What about patients who are brought into trauma care or who are unconscious, when we don't know who the patient is? We can design the system to be open and support situations as they happen.

What about people lying when they update their records? We need accurate provenance of data. It is hard to encode trust in the data, but you can trust the data as long as you have accurate provenance. Data other than time-series data can be difficult to visualize, but technologies like Wave can display the flow of information. We should also understand that the user interface is different for different consumers and needs to be integrated.

Even lawyers have different interpretations of health policies. Implementing these policies into code is a non-terminating problem. Pharmacies need the patient's name and prescription, so would presenting an anonymized token be sufficient? Pharmacies don't need your name; they can authenticate you if the barcode you present matches the barcode on the prescription. But rules could change in the real world if the patient forgets to bring the barcode with her or when the pharmacist looks at the medicine and understands what the patient's medical condition is.

The system could function very well if we determine what information needs to come together to perform the function we need. Note that we might not be able to understand who needs what from the raw information that we get. Consider the digital cash argument, where you could perform transactions without giving away too much information. How is it different in healthcare just because you deal with medical information and insurance companies? Those are similar but we have to be more careful, because a failure in the system could prove fatal.

When coming up with technologies, should you try to envision environments outside of North America? During floods in Pakistan, the government provided aid only if you provided your ID card. It would be interesting to develop technologies to work with antagonistic consumers.

Insurance companies need to know how a patient is doing. How can they assess risks and charge premiums when a patient's health data is anonymized? We know that they need information about everybody, but is it their right to have all the information they want? Chase discussed the purpose and use limitation in her talk. What are some potential applications in research settings? It is difficult to guarantee that the data you give out is used only for a given purpose. But it is possible to prove that a particular statement that you claim to be true is true, rather than giving out data and trying to protect it from being used elsewhere, which is a hard problem.

How do you see primary care physicians using such technologies when they have no IT staff? People will trust technologies that allow them to partition health information and protect it. Doctors view the role of PHRs as supplemental, equivalent to clipboards. They can take data in the PHRs and do their diagnosis. Patients can do lots of stuff with the data as well. PHRs will be adopted slowly, one step at a time. As a first step, we can reduce redundant procedures and tests—for example, when you redo a test with another doctor because you can't transfer the fact that test was done earlier.

Central healthcare repositories are not possible in the US, since we don't have the technology to manage access control. If PHRs are starting to fill with information collected from glucometers or insulin pumps, how does the patient know whether to believe the data? We need to look at higher-level abstracted data, rather than looking at the raw data. HealthVault digitally signs data before uploading it. But the processing power of some medical devices is not ready for encryption to support the provenance claims. So you could still attack between the device and client PC, unless you are timing the data on the device itself. It is hard to determine what is enough: is it good enough for the client application to sign the data? You have no option but to trust the outcome of the device, unless the device is broken. The other issue is how you get data into PHR—issues such as, is it my glucometer or not? There are good techniques to authenticate devices that have no I/P or O/P.

Information from medical devices has to be summarized. Also, you can't have a machine diagnose like a physician. If we dump raw data, no one is going to look at it. EHRs are crucial not to just repeat tests but to do tests. Health providers have not seen any patients who use PHRs, even though they have treated students and other technologically advanced patients. Another bit of information that could be captured by PHRs would be a list of a patient's medications, which patients never remember; so it would be good to have bi-directional contact with pharmacies. Adoption of PHRs is slow since there are no central repositories to get information from. We need to make deals with organizations to get data flowing. Also, there are no computers that patients could use in a doctor's office. Adoption will happen gradually with time and education. PHRs are at an early adoption phase. The most practical thing happening now are HIEs. PHRs have been left behind so far.

4th USENIX Workshop on Offensive Technologies (WOOT '10)

August 9, 2010
Washington, DC

VULNERABILITY ANALYSIS

No reports are available for this session, which included the following paper and invited talks:

- ***All You Ever Wanted to Know About Dynamic Taint Analysis and Forward Symbolic Execution (but Might Have Been Afraid to Ask)*** (Invited Talk)
  Edward J. Schwartz, Thanassis Avgerinos, and David Brumley, Carnegie Mellon University
- ***Zero-sized Heap Allocations Vulnerability Analysis***
  Julien Vanegue, Microsoft Security Engineering Center
- ***Beyond Heuristics: Learning to Classify Vulnerabilities and Predict Exploits*** (Invited Talk)
  Mehran Bozorgi, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker, University of California, San Diego

CRYPTOGRAPHY, ETC.

Summarized by Adam J. Aviv (aviv@cis.upenn.edu)

- ***Recovering Windows Secrets and EFS Certificates Offline***
  Elie Burzstein, Stanford University; Jean Michel Picod, EADS

Elie Burzstein discussed the Windows Data Protection API (DPAPI), a "black box" for encrypting and decrypting data that is used in many different parts of the Windows operating system, including the Encrypted File Systems (EFS), as well as a variety of other programs (Skype, Explorer, WiFi, etc.). Burzstein provided key insights into mounting the Windows EFS on Linux. This work also shows how one may perform a key escrow attack on the DPAPI to achieve this goal.

In the first part of the presentation, Burzstein introduced the DPAPI in great detail. Moving quickly, the talk covered the ins and outs of key management and the various structures that store and implement encryption. Some of the more important points are that the keys used by the DPAPI are seeded with a hash of the user's password, and keys renew every three months (and when the password changes).

The current keys are stored in the %APPDATA%, a Windows protected file, inaccessible by outside applications or operating systems. Still, Windows must know which block is encrypted with which key, and to do that, timestamps are used. This is where things get interesting. Burzstein et al. noticed that the timestamps can be altered to prevent key renewal. However, there is still the pesky password-changing problem, but the master key describes the current password hash. A password chain is used, so by using the current password it can decrypt all previous encrypted blobs.

Burzstein could not demo his tool because he was presenting from a Mac. He did outline some goals: to make this work on Windows 7 and to look at retrieving the password from non-volatile memory. "Questions?" he asked in conclusion. "The best part of the talk is I never get questions."

- ***Crawling BitTorrent DHTs for Fun and Profit***
  Scott Wolchok and J. Alex Halderman, University of Michigan

In a fascinating presentation, Scott Wolchok discussed crawling the BitTorrent DHT (for fun and profit). This work is closely related to Wolchok's widely publicized work on Un-Vanish (NDSS '10) where he crawled the Vuze distributed hash table (DHT) to defeat Vanish (Sec '09). In this work, Wolchok concerned himself with the primary purpose of the Vuze DHT: to catalog BitTorrent (BT) meta-information, and what can be done with information collected from crawling this data.

Wolchok began his talk by noting that torrent-tracking Web sites are under legal attack because of their centralized nature. As a result, distributed and decentralized tracking services are quickly becoming the norm. Such services make use of DHTs, and the ability to crawl the DHT to collect torrent information could be seen as a defense against legal attack. Although a torrent site may be taken down, a single, overnight crawl of the DHT provides enough information to rebuild the BT site. Conversely, the same crawl also reveals a large amount of information about the users who download torrents, which may be used to file lawsuits—fun and profit.

Perhaps the most interesting part of the talk was when Wolchok presented the results of his crawls with respect to the variety of different torrents seen: "Everything in the top seven infringes copyright. It isn't used to download Linux ISOs." The top 1,000 torrents were also "not obviously" copyright neutral, and most torrents tended to be fairly recent TV and movies. Wolchok bemoaned that he had to stop the crawl prior to the *Lost* finale (having only done a single crawl, resulting in an estimated 20% coverage). The most recent *Lost* torrent was one of those popular torrents and its activity seemed to spike on Friday night and Saturday morning. He offered one explanation: "Pirates have jobs too," and probably don't get around to downloading the show until the end of the work week.

- ***Practical Padding Oracle Attacks***
  Juliano Rizzo, Netifera; Thai Duong, VNSECURITY

Juliano Rizzo demonstrated how he and his co-author, Thai Duong, performed online attacks by altering the CBC padding in captured blocks of ciphertext (first presented by Vaudenay at Eurocrypt 2002). The key observation of Rizzo and Duong is recognizing that *padding oracles* are everywhere on the Internet, which allows an attacker to crack encrypted cookies, CAPTCHAs, and much other encrypted content. By slightly altering the padding bits of the encrypted blocks sent to the oracle, a response of either "Invalid" or "Valid" is enough to decrypt one byte of the ciphertext.

Repeating the process, the message is incrementally decrypted, back to front.

The most exciting part of the presentation was when Rizzo played some videos of his padding oracle in action. In the first demo, an encrypted cookie from a JavaServer Faces client was incrementally decrypted. In the second demo, Rizzo showed how this attack can be used to break a CAPTCHA. The text of the CAPTCHA entry is encrypted with the CAPTCHA image, and the padding oracle is the CAPTCHA server. Slight alterations of the padding region and a useful error message from the server ("PADDING ERROR") are more than sufficient to decrypt the CAPTCHA text. Again, the video demo presented was very cool, and slowly but surely the text of the CAPTCHA was revealed within the tool's display region. (The video included a text flash "10 minutes later," which got chuckles from the audience.)

THE WEB AND SMARTPHONES

Summarized by Scott Wolchok (swolchok@umich.edu)

- ***Busting Frame Busting: A Study of Clickjacking Vulnerabilities on Popular Sites*** (Invited Talk)
  Gustav Rydstedt, Elie Bursztein, and Dan Boneh, Stanford University; Collin Jackson, Carnegie Mellon University

Collin Jackson opened by explaining that frame busting refers to JavaScript code that Web sites use to prevent themselves from being framed, and that Web sites typically don't do frame busting very well. He explained that frame busting code typically consists of two parts: a conditional statement intended to detect framing, and a counteraction intended to remove the framing or disable the page. Frame busting is intended to defend against several attacks, including clickjacking, where attackers overlay a benign-looking page intended to trick a user into performing some action on a victim page (e.g., deleting a Twitter account), and attacks on per-site images attempting to authenticate a site to the user as a phishing defense, which can be defeated by framing the victim site's login page to display the image.

Jackson then presented the results of the authors' survey of frame busting code on the top 500 Web sites according to Alexa. They found that frame busting was very common on the top 10 Web sites (60%), but not so common on the top 100 (37%) and top 500 Web sites (14%). They also observed that frame busting code was very diverse, with at least 10 different conditional statements and even more different counteractions, and Jackson claimed that every site in the top 500 had broken frame busting code. He elaborated on problems with specific sites, most of which revolved around attempts to allow framing from certain referrers.

Next, Jackson covered a variety of other attacks on frame busting code. Location clobbering attacks browser bugs that allow a framing site to mask top.location to prevent the framed side from detecting framing. The attacker can also "ask nicely" in JavaScript to get the user to cancel the framed page's redirection counter-action and can cancel the navigation programmatically by overloading the browser with 204 No Content responses. Several browsers allow disabling JavaScript in an iframe, and reflected XSS filters in IE8 and Chrome can be abused to remove frame busting scripts as well.

Jackson closed by discussing mitigations to the frame busting problems he presented. A pair of HTTP headers, X-Frame-Options and Content Security Policy, allow sites to control framing at the HTTP level, although they have tradeoffs in terms of complexity and flexibility. The authors put forward a new JavaScript frame busting defense that "fails safe" by rendering the document invisible if it is unable to frame bust.

Rik Farrow asked where the defense code could be found, and Jackson replied that it is located at http://seclab.stanford.edu/websec/framebusting/. Someone asked Jackson to clarify whether JavaScript had to be mandated to protect against frame busting attacks. Jackson stated that this is somewhat true and very controversial, as many people think that JavaScript is evil and a security problem. Jackson said that he is skeptical of solutions that try to lock down Web features, although he recognizes that some people may want to. He stated that the X-Frame-Options header works with JavaScript disabled. Someone else asked how to protect a user like his mother, who would click everywhere on the invisible document generated by the authors' frame busting code in an attempt to fix the "problem." Jackson responded that the code sets the display:none CSS property on the body element, which prevents click events.

- ***Smudge Attacks on Smartphone Touch Screens***
  Adam J. Aviv, Katherine Gibson, Evan Mossop, Matt Blaze, and Jonathan M. Smith, University of Pennsylvania

Adam Aviv opened by summarizing the authors' work: "I took a lot of pictures of smartphones with smudges on them." He presented several examples of forensic information leakage, including taking a rubbing from a pad of notepaper, wear patterns and residual heat on keypads, and residual fingerprints on a touchscreen. He explained that the authors' work focused on smudges left on Android phone touchscreens after performing the password wipe sequence to unlock the phone. In the wipe sequence, the phone shows a grid of nine points, and the user must trace a line through several of them. Points can neither be skipped nor reused. Aviv observed that the pattern space is fairly small; it consists of 389,112 patterns, and a similar PIN entry space (4–9 digits, used once) contains over 1 million passwords.

Next, Aviv explained the experiments that the authors performed. They considered one particular swipe pattern touching all nine dots and selected to provide several different directions. The swiped phones were photographed while varying the lens angle and the vertical angle to the phone. The photographs were classified on a scale from 0 to 4,

where 4 was fully observable and 1–3 were partially observable with the loss of one or more directions.

The first experiment dealt with determining the angles and lightings that provided for ideal collection of smudges and used four smudge configurations: an HTC G1 with "normal" touches, "light" touches, touches with facial contact, and a Nexus One with "normal" touches. After Aviv showed a photo of the four experimental configurations, Rik Farrow interrupted to ask about the classification of each swipe pattern; Aviv responded that everything in the photo was a 4, but the projector was not rendering the photo faithfully. He also mentioned that the classifier is allowed to change the contrast of the photo in software. The experiment found that putting the phone up to the face caused a large smudge, and then entering the pattern cleaned the phone. Thus, facial contact yielded the highest retrieval rate, whereas light touches had the worst, although 37% of such photos gave at least some information about the pattern. Aviv displayed a photo illustrating that directionality of the swipe is visible because of swipe overlays at the corners; one can see which direction is on top.

The second experiment dealt with two types of simulated application usage prior to the swipe: dots due to presses of numbers or other taps, and streaks due to swipes. The worst case was when the phone is touched everywhere. Aviv also pointed out that recovery is much better when the pattern is entered after application usage instead of before, as one might expect. The third experiment dealt with two incidental clothing contact situations, both of which degraded or lost directionality information while not completely occluding the swipe.

Aviv closed by considering further work, including research into the human tendency to choose passwords with low entropy. He observed that because of the small password space, a small amount of partial information, such as a dictionary and a smudge, might be able to reduce the space below the 20-guess threshold. For example, removing passwords that include a hard-to-enter 30-degree stroke (e.g., from 1 to 8 in the standard 3x3 telephone keypad layout) reduces the pattern space by 50%.

Aviv's presentation inspired many questions. Someone asked why both the Nexus One and the G1 were included, to which Aviv responded that one screen is glass and the other is plastic. A second audience member said that it seems obvious that smudges might leave password swipe information, and asked whether there are any other phone applications where users might leave information. Aviv responded that the iPhone PIN is somewhat similar. He admitted that the iPhone on-screen keyboard is too small for smudge attacks, but speculated that iPads might be vulnerable. He closed by saying that it is difficult to determine the order of keystrokes from a screen full of on-screen keyboard smudges, unlike residual hot spots on a keypad. A third questioner asked whether people post pictures of their Android phones on Flickr. Aviv responded that one person posted a picture of a phone on a blog and asked if his pattern was discernible, and added a disclaimer that the authors were not the first to think of this attack, but they were the first to perform a systematic study. The questioner clarified that he was interested in accidental phone posts, to which Aviv replied that not many people take pictures of their phones. A fourth questioner asked how thoroughly phones had to be wiped to remove smudges. Aviv's reply was that it is fairly hard and that he found that two wipes were often necessary, and clarified that the focus of the study was whether a random picture would be able to view smudges.

Another audience member asked whether application developers could require the user to enter a random sequence to generate a random smudge as a mitigation. Aviv replied that such a solution might work, but it's putting the burden of fixing a bad security design on the user. He also said that the paper's reviewers asked for solutions to the problem, but he did not have any good solutions. He suggested numbering the dots and changing their order so as to change the pattern, but that would add 30-degree swipes. Another audience member suggested using a smaller keypad and shifting its location on the screen, but Aviv said that refocusing the camera would counter that defense. Scott Wolchok asked about the extent to which guessing the password is the easiest way to gain access to a phone, as opposed to exploiting some software vulnerability or developer access. Aviv replied that such an exploit was outside the scope of the authors' work, and noted that smudge attacks were applicable in scenarios other than finding a lost phone: an attacker might be surveilling a target, notice that the target's phone was smudged, and quickly steal the phone to recover information before replacing it. Aviv was then asked whether different screen covers (matte or glossy) mattered; he responded that dark screens would be better for security.

- ***Framing Attacks on Smart Phones and Dumb Routers: Tap-jacking and Geo-localization Attacks***
  Gustav Rydstedt, Baptiste Gourdin, Elie Bursztein, and Dan Boneh, Stanford University

Baptiste Gourdin spoke about attacks on mobile phone Web browsers. He highlighted key differences between mobile browsers and traditional browsers: the attacker can zoom to the element of his choice and easily remove browser chrome by scrolling the page down. Gourdin included a demonstration that used JavaScript to scroll down and remove the true chrome while displaying a spoofed chrome, including an SSL security indicator.

Next, Gourdin discussed tapjacking attacks. He began with a demonstration clickjacking attack on Twitter that overlaid the permanent account deletion page on top of the play button for the "BEST GAME EVER." He briefly discussed mitigations such as frame busting, but said that frame busting can crash or fail on mobile browsers. Moreover, click-

jacking protection is even rarer on mobile sites, so tapjacking amounts to "clickjacking on steroids." Gourdin provided a mobile version of his demonstration attack on Twitter and said that the vulnerability had been fixed, but was previously a live mobile Twitter vulnerability.

In the second part of his talk, Gourdin presented applications of frame leak attacks for stealing private data. He began with Paul Stone's scrolling attack from Black Hat, which allows the attacker to violate the same-origin policy and determine whether an anchor is present in a page, by placing a hashtag of the form #foo at the end of a framed URL and testing the frame's scroll position. He demonstrated how the attack could be used on Yahoo Mail Mobile to determine whether a victim received mail from a particular sender. He also pointed out that Facebook's clickjacking defense, a large dark div overlaid over the page, does not prevent frame leak attacks. Thus, an attacker can test whether a user is logged in by searching for the registration form, and can also determine which user is logged in. Facebook fixed this vulnerability by simply displaying a Facebook logo when framed, rather than showing information behind a div.

Bill Cheswick asked if the iPhone's button (used to quit the browser) mitigates these attacks. Gourdin replied that it would certainly quit the browser, but the user would still be attacked whenever he visited attacker.com.

AFTER YOU GET EIP

Summarized by Scott Wolchok (swolchok@umich.edu)

- ***Interpreter Exploitation***
  Dionysus Blazakis, Independent Security Evaluators

Dionysus Blazakis said he would show why exploit mitigations are only a safety net and vendors still need to remove bugs. From an academic point of view, he provided an example of a non-trivial information leak and showed why the leak is an emerging class of bugs. He urged academics to attempt to formalize how to find such bugs.

Blazakis discussed data execution prevention (DEP) and address space layout randomization (ASLR) and how they complicate attacks. The combination of the two makes attacks difficult, because DEP allows return-oriented programming and return-to-libc attacks, but ASLR makes such already difficult attacks probabilistic at best. An attacker looking to circumvent this combination might use information leaks and heap spraying in order to obtain executable pages with known or easily guessable locations. Blazakis then introduced two techniques, pointer inference and JIT spraying, that can be used to bypass existing exploit mitigations. The pointer inference technique interacts with the object structure of the Tamarin VM used by Flash to generate native code, in which the least significant bits of values (called "atoms") are used to encode type information. Objects are stored as tagged pointers, but integers and other primitive types are stored by value. Tamarin's general-purpose hashtable maps atoms to atoms and can be iterated over in hash order. Blazakis's insight is that the table uses the values themselves as the hash, so mixing integers and objects in the table results in integers being compared to pointers, which leaks address bits. In particular, he stated that one can determine whether an address is even or odd by putting it into two tables filled with even and odd integers and determining in which table the pointer doesn't collide. Someone asked how many bits were leaked, and Blazakis responded that about 25 bits of a 32-bit pointer could be recovered. However, the information leak is just some arbitrary heap address; there are controllable fields, but the leak is not directly exploitable.

Blazakis then moved on to JIT spraying, his second attack. Rik Farrow pointed out that JITs write code to the heap, and the pages with code have to be marked executable. Blazakis continued by explaining that a long XOR expression in ActionScript will cause the JIT to generate a compact x86 instruction stream consisting of MOV and XOR instructions, and stage-0 shellcode consisting of 2-byte instructions can be encoded into the constants manipulated in the expression. The emitted function can also contain a pointer to a string constant used to host stage-1 shellcode. Generating many such functions will effectively spray the ActionScript heap with shellcode. Blazakis demonstrated his exploit, which took about a minute.

Someone asked if these attacks meant that he had to eschew JIT programs to remain secure. Blazakis responded that in short, the answer was yes.

- ***A Framework for Automated Architecture-Independent Gadget Search***
  Thomas Dullien and Tim Kornau, zynamics GmbH; Ralf-Philipp Weinmann, University of Luxembourg

Tim Kornau spoke about the goal of using return-oriented programming tools across multiple platforms. He enumerated the common architectures today and stated that exploits should run even on a refrigerator. Specifically, the authors' goals are to execute code in the presence of the NX bit and when binaries are signed, but circumventing ASLR is outside the scope of the talk. The strategy the authors adopted was to reuse application code (i.e., through return-oriented programming) without relying on returns or return-like instructions; rather, they intend to extract semantic information from the binary. Kornau then introduced REIL, a 17-instruction RISC instruction set where all instructions are three operands and have no side effects. REIL is currently unable to support exceptions, floating-point instructions, or 64-bit computing, but those capabilities are under development. Someone asked why exception support was important; Kornau responded that, for example, MIPS's integer instructions use exceptions to represent various things, and it's difficult to model exceptions architecture-independently.

Kornau then explained the algorithms developed by the authors. In the first stage, data is collected from the binary by first extracting REIL expression trees from the native instructions and then extracting path information by bottom-up depth-limited search from the end of the gadget. All paths are stored in the same expression tree by multiplying the condition bit together with the operations. In the second stage, the expression trees for single native instructions are combined along paths and simplified (e.g., by constant folding). In the third stage, the authors locate useful gadgets by using a tree match handler determining whether a condition is met for each needed operation. The algorithm selects only the simplest gadget for each operation. Kornau stated that the algorithms are currently functional, but searching for gadgets is highly platform- and compiler-dependent. He cited difficulties like branch delay slots (MIPS), predicated execution (ARM), and register windows (SPARC). Further work includes an abstract gadget description language, an automatic gadget compiler, more platforms for REIL, and better understanding of the implications of different compilers.

Rik Farrow clarified that by "gadget" Kornau meant "a block of code that does something." Kornau replied that yes, traditionally, it does something useful and must be chainable to other gadgets. He stated that the authors' analysis differs from the traditional return-oriented programming analysis because it does not reason about unintended instructions and requires a valid disassembly up front. In reply to a second question, Kornau stated that fuzzy tree matching only searches for certain operands, because REIL has a very normal structure. A third audience member asked how large binaries had to be in order to find Turing-complete gadget sets. Kornau replied that it was very binary-dependent; he cited libsystemb as an example that generated over 240,000 gadgets and said that an attack can usually be adapted to such large binaries, whether or not the gadget set is Turing-complete. The questioner then asked how large the files were in bytes. Kornau said that he believed that libsystemb is about 200KB, but he was not certain.

- ***English Shellcode*** (Invited Talk)
  Joshua Mason and Sam Small, Johns Hopkins University; Fabian Monrose, University of North Carolina at Chapel Hill; Greg MacManus, iSIGHT Partners

Sam Small discussed how English shellcode can be used to avoid network intrusion detection systems (NIDS). He began with a review of shellcode and evading filtering and detection; shellcode transformations have been used previously to bypass application-level input filters, but, arguably, not to evade detection. He stated that NIDS works by using either regular expressions and signatures or emulation. The problem with emulation is that the attacker can use domain-specific knowledge of the application, such as registers or memory, and eflags in particular are almost always reliable. Thus, NIDS can't be aware of which paths are actually taken in a particular string, and the attacker can set eflags using arithmetic operations if necessary. Moreover, the attacker can use self-modifying code, even if he is constrained to English.

Small then moved on to the details of English shellcode generation. English shellcode has three parts: the pre-decoder, the decoder, and the transformed shellcode. The decoder unpacks the transformed shellcode, but cannot be written in English (because of instructions like lods and jnz), so the English pre-decoder is included to unpack the decoder. Small stated that the decoder would not be explained in the talk and moved on to the details of the generation engine. The language generator is based on beam search and uses a large corpus of text to build a language model. It looks at every word in the corpus that could follow the current word in the shellcode, concatenates it with the current shellcode string, and, using a scoring engine, determines how well the modified string accomplishes the desired code. The engine is in two parts: the sentinel breaks the shellcode into chunks of instructions and passes them to the executor, which it monitors through ptrace. The sentinel eventually returns a score. Small stated that the proof-of-concept system took about 12 hours, but combining the sentinel and executor into one process through a "feat of engineering" reduced the time to 20–30 minutes.

Small closed by showing some samples of English shellcode, including two quite long encodings of exit(0). The sample text, while not entirely "readable," contained many coherent phrases and popular topics. Small pointed out that the letter "r" is a jump and can be used to skip more English-like blocks of the shellcode paragraph.

Someone asked what the average size increase of English shellcode was. Small responded that there are several factors, but it is easily over 100x, which isn't prohibitive if shellcode can be placed on the heap. He said that the size increase depends on several tunable parameters that have not yet been tuned for space. Someone else asked whether the generated shellcode was contextual, as Small mentioned at the start of the presentation. Small replied that it sometimes was, and could avoid choosing instructions that access memory and registers with unknown values. A third audience member suggested that an online game could be used to get people to write sensible text to fill in the shellcode, and Small mentioned that his co-author would often ask for words that fit certain constraints during development. Someone else asked about searching for code in standard texts, such as help files. Small replied that such searching is theoretically possible, but it seems very difficult. A fifth questioner asked how much of the pre-decoder was predictable, and Joshua Mason replied that it is specifically designed to make prediction impossible. If a NIDS matched on the necessary bytes, it would also block valid text.

New Security Paradigms Workshop (NSPW 2010)

September 21–23, 2010
Concord, MA

Summarized by: Matt Bishop, University of California at Davis; Steven Greenwald, Independent Consultant; Michael Locasto, University of Calgary

The 2010 New Security Paradigms Workshop began with a reception and dinner on September 20 and ended at noon on September 23. The highly interactive workshop has its participation limited to about 35 people. NSPW encourages authors "to present ideas that might be considered risky in some other forum, and all participants are charged with providing feedback in a constructive manner. The resulting intensive brainstorming has proven to be an excellent medium for furthering the development of these ideas." For more on NSPW, check out http://www.nspw.org.

Disclaimer: due to severe time constraints, the presenters have not checked the accuracy of these summaries. We take full responsibility (but not the blame) for all mistakes.

PANEL

- ***Why Is There No Science in Cyber Science? (first part)***
  Panelists: Roy Maxion, Tom Longstaff, and John McHugh
  Moderator: Carrie Gates

The theme of this panel was the relationship of science with computer security. All three panelists believed that computer security would benefit from a good dose of scientific rigor.

John McHugh began by polling the audience to see how many believed scientific methodology should be applied to experimental computer security research. Two thought the claim was questionable; everyone else agreed with it. McHugh pointed out that injecting this rigor requires clear statements of hypotheses (which is easy), and then collecting data and performing data analysis in a way that can be reproduced (which is hard). In the discussion that followed, someone pointed out that perhaps the classical model of experimentation in physics is a poor analogy to experimental computer security, because the universe is benign and will not lie to you. Several people suggested that a better model is anthropology, noting that field workers who collect the data and lab analysts who analyze and draw conclusions from those data are completely different. This is like the culture of computer security.

Roy Maxion followed. He pointed out that making computer security research more rigorous could be done incrementally. To emphasize the importance of describing the data collection methodology, he cited a study on using mouse movements to identify users. The researchers built a special apparatus and had 11 people browse. The data they collected enabled them to identify each person. But they did not tell people *what* to browse. That the subjects were browsing different Web sites might have produced the discrepancies that enabled the researchers to uniquely identify each person. In the discussion that followed, someone pointed out that science is not necessarily hypothesis testing, but is really contributing to generalized knowledge—and this means that papers that describe attacks are usually not scientific, as they are not generalizable. Maxion agreed, commenting that problems often arise when one does an experiment using a small sample size and tries to generalize the results to a large population. In response to another comment that the problem is the lack of an argued methodology for phenomena we wish to investigate, Maxion recommended that researchers have a method for doing something, and the method be made transparent by including in papers an experimental methodology section detailing exactly how the thing was done.

Tom Longstaff went last. He discussed the culture of publication, lamenting that there are now so many venues for publication that conferences and workshops often accept weak papers because they need to fill sessions. Further, program committees often take papers that are not good science, and instead of rejecting them on those grounds, try to fix them. He argued that we need to reject such papers outright. He concluded with a challenge to the workshop. He noted that papers in the workshop fall into two categories: speculative papers putting across new ideas, and papers with conclusions and results. He asked whether the former provided ideas that could be rigorously evaluated so that a good scientific paper could be produced, and whether the latter had scientifically sound methodologies and conclusions. In the ensuing discussion, one participant suggested focusing not on "science" but on "theory building," arguing that the methods used in physics are different from those in social science, but theory building applies to all disciplines in that it requires identification of premises and rules for drawing conclusions (whether inductively or deductively). The response was that the panel was discussing experimental science in the way it handles and manipulates data being collected, and so can resemble social science as well as physics. There was also considerable discussion about the relationship between engineering and science, with a comment that security as a discipline does not know what it is trying to do—we want to make things "secure" but do not know how. The response was that it's a bit like working on a perpetual motion machine; we can't get there, but we can continually approach the goal, and we can measure how far we fall short of it. People expressed hope that security could measure how far it falls short.

All three panelists emphasized the importance of including a section on methodology, especially experimental methodology, in all papers so that reviewers and readers can assess the results properly.

- ***E unibus pluram: Massive Scale Software Diversity as a Defense Mechanism***
  Michael Franz

Michael Franz reviewed the process of software creation and distribution. The current practice is to use a "unicompiler" that produces a single object code, so all instances are susceptible to the same attack. He proposed a "multicompiler," which gives different object files to different customers; the same attack would not work on all object files. He proposed doing this diversification in an "app store" to simplify the distribution process, hiding it from both the developer and the user so that their processes need not change. This solves the problem of attackers reverse-engineering patches to find vulnerabilities, because the store would either need to send each customer a custom patch for their version of the application or simply send a new, patched version of the application. He discussed four paradigm shifts that underlie this work: online software delivery, ultra-reliable compilers such as just-in-time compilers, cloud computing, and the economy of scale.

Franz noted that there were two costs involved. The first was the cost of generating the diverse object modules. As a compiler generates object code by choosing from several possibilities, it could simply save all those possibilities and the app store could choose one set to generate the software to send to the customer. Some users would get non-optimal software; when asked about this, Franz said he did not know the impact of this or how much degradation would be involved. Then there is an up-front cost of actually distributing the software. Assuming $0.18 per hour for cloud computing, Franz estimated that each unique version of Firefox, which has 30 million lines of code, would cost $0.09.

In the discussion, someone pointed out that use of the app store to generate the diversity solves many problems but introduces a single point of failure. Someone also asked whether redistribution would need to be banned, and Franz replied that he didn't care: the number of diverse object files would make attacking much harder even in the face of redistribution. A number of people, including Franz, emphasized that this method works better as the scale increases, and if done on a small scale rather than a large one, the benefit would be minimal. Questions were raised about the amount of diversity; this is one of the research areas that must be explored. For example, the amount of diversity would control whether one could generalize from a patch to be applied to a single software instance to an attack that would work across all patches.

- ***On Information Flow for Intrusion Detection: What if Accurate Full-System Dynamic Information Flow Tracking Was Possible?***
  Mohammed I. Al-Salah and Jedidiah R. Crandall

What if we used information flow tracking methods for intrusion detection instead of (or in support of) the currently popular appearance-based or behavior-based methods? If we did, then we should research ways of approximating dynamic information flow tracking as accurately as possible. This also means that we should move to the paradigm of looking at global properties and to dynamic quantitative flow analysis.

Jed Crandall described their use of the dynamic information flow tracking (DIFT) method, which tags/taints data in order to measure the information flow throughout the system. As a first step towards this goal, they created a prototype DIFT system that supports address and control dependence in a general way and measures these specific information flows.

A lively discussion ensued. At one point, Crandall emphasized that, because they have only a prototype system, they did need more accuracy and made a lot of approximations. He viewed their biggest research challenge as how to handle the expansion of taint. Regarding the quantities of information passed, Crandall mentioned that they looked at data provenance and forensics and they now look at threat models. They also want to have a system where they can make statements such as "These data first were cut-and-pasted from Office and saved to a text file and then were saved to a USB memory device"—in such a case everything becomes tainted but just a little bit at a time, so (as an example) labeling it with bits from a tainted file might wind up transmitting something like 0.00005 bits from the contaminated file.

- ***A Stealth Approach to Usable Security: Helping IT Security Managers to Identify Workable Security Solutions***
  Simon Parkin, Aad van Moorsel, Philip Inglesant, and M. Angela Sasse

Simon Parkin presented a stealthy approach to convincing IT security managers and chief information security officers (CISOs) to include usability in their policies. The tension between security and usability often makes people believe that the two are incompatible, so they sacrifice usability for security, but research has shown that, in general, the two can be compatible. CISOs, however, typically do not know how to apply current research in usability. This work engaged CISOs by using their language and methods, testing the ideas by involving them in a user-centered design. They led three CISOs from large organizations through a semi-structured requirements analysis for a password policy. The researchers developed a tool that helped show the impact of various parameters such as length, complexity, change notification, and other aspects of passwords on the productivity of workers, the cost of the control, and the number of breaches of security. They did mock-ups of the tool's output and met with the CISOs to see if they could integrate usability issues into their existing processes.

The discussion was lively. The researchers picked a password policy to begin with; someone asked whether they included organizational processes such as auditing and so forth. This led to some comments that the choice of a user-centered activity approach rather than a user-centered design might be more fruitful, because while CISOs care

about users, they often do not know much about what users are doing. A recurring question concerned the background of the CISOs in the study. In particular, what did the CISOs think was most important? This is based in part on their evaluations, the criteria for which varied widely depending on the business. One CISO said that productivity did not count but that the lack of security breaches did.

This also led to a discussion of assumptions. The study focused on CISOs, but did not examine other parts of the organization, nor did it gather data from anyone other than the CISOs. The presenter responded that gathering data from others was the next planned step in the research, and by talking to the CISOs, they learned which stakeholders they needed to talk to later. They plan to incorporate this information into their tool, which prompted another observation that senior management often wants to hold back information from subordinates, so they may not be able to include it.

The session concluded with a suggestion that some day the tool might be able to replace CISOs entirely. Someone observed that CISOs have two roles: making the policies, and being fired when there is a breach. The conclusion was that, assuming the tool could incorporate artificial intelligence techniques and be generalized to include everything, it might be able to replace the CISO—but this would be very bad for users.

- ***VM-based Security Overkill: A Lament for Applied Systems Security Research***
  Sergey Bratus, Michael Locasto, Ashwin Ramaswamy, and Sean Smith

Michael Locasto and his co-authors challenged the common idea that virtual machines are isolation and introspection panaceas and, in particular, that any credible research into kernel-level modifications requires the use of a virtual machine because one can only monitor software effectively from a lower layer. He argued that emergent complexity in the virtualization environment greatly increases the attack surface. There are pressures from below (such as remote managers, and the need to maintain both the host operating system and the virtual machine operating system), at the interface (the need to create and maintain guest-host APIs so the host can extract data from the processes running on the virtual machine), and from the virtual machine itself (the information flow policy and the machine's use as a resource emulator and controller).

Locasto suggested that the key research challenge in this space requires devising mechanisms that monitor systems at the same privilege level rather than from below. These "self protection" mechanisms still represent an interesting path of research. Indeed, in some scenarios, software must monitor itself.

Someone pointed out the "observer effect," in which the very act of introspection of a virtual machine taints the guest operating system, and asked if there were another way to get the information. But this paper was simply identifying a semantic gap between what is being monitored and observed. Determining the best mechanism to extract the information is a fundamental challenge and one the authors did not solve. Another part of the discussion brought out the fact that, originally, virtual machines were designed for multiplexing separate systems onto one piece of hardware. But when the hypervisor uses the system hardware for paging, for example, it no longer acts as an intermediary between the guest operating system and the host operating system. While this improves performance, it also greatly increases complexity because now new traps and checks must be added to support the security requirements. Thus, there is a gap between what we want and what the hardware provides. This led to some thoughts about a lightweight virtualization mechanism, which the presenter said was a valid approach but not considered, because the research focused on detecting rootkits.

Locasto concluded with the observation that as our dependency on virtual machines increases, they become less trustworthy as they become more trusted.

- ***A Billion Keys, but Few Locks: The Crisis of Web Single Sign-On***
  San-Tsai Sun, Yazan Boshmaf, Kirstie Hawkey and Konstantin Beznosov

San-Tsai Sun explored the lack of acceptance of single sign-on mechanisms on the Web. Their model consisted of three parties: a user, an identity provider (which manages the single sign-on credentials), and a relying party (which should accept and trust the credential). The relying party has no business incentive to rely on the identity provider, because the relying party is responsible for any loss when the identity provider is compromised or unavailable. Further, relying party sites often rely on user data to survive; not obtaining that data, by relying on the identification by the identity provider, may not be acceptable from a business point of view. The user has no incentive to rely on the relying party's use of the credential, because the different user interfaces among all parties that rely on the identity provider is confusing. Perhaps more importantly, the goal of single sign-on is to simplify the identity management process for the user. But most browsers come with password managers, hiding the complexity of identity management. Further, the use of an identity provider creates a single point of failure: if the identity provider is unavailable, the user cannot get her credential. Finally, privacy concerns abound.

The model was praised for its structure and thoroughness. Someone suggested that the authors should integrate other sorts of architectural impacts and error handling, and much of the discussion that followed concerned these points. The work here focused on individuals, not organizations, so a relying party for businesses rather than individuals may have different issues. Someone suggested that, intuitively, the use of single sign-on improves both security and usability, but no research was cited to support this view, and others disagreed. For example, the use of an identity provider enables a denial of service attack against the user;

the identity provider (who in many cases is invisible to the user) can void the user's credential at will. This led to a discussion of federated identity providers, including the comment that relying parties determine which identity providers they trust. Participants noted that several large corporations and, especially, governments were developing their own identity providers; for example, the German government will provide a single sign-on that makes a verified shipping address available to e-merchants, simplifying the process of purchasing items over the Internet. In addition to browsers providing password managers, many Internet service providers do as well. These two observations vitiate the business case for single sign-on because the ISPs and browsers provide the same level of convenience to the user that an identity provider would.

The discussion concluded with an ancient observation that provides a basis for much of the work: who benefits? What benefit does the single sign-on provide for all?

- ***To Boldly Go Where Invention Isn't Secure: Applying Security Entrepreneurship to Secure Systems Design***
  Shamal Faily and Ivan Flechais

Shamal Faily described the goal of this work as applying models and principles to create, organize, and manage security design elements in such a way as to improve system security. The authors compared three different models of innovation, and contrasted security architects with security entrepreneurs in each. Their models of innovation were incremental vs. radical, component vs. architectural, and static vs. dynamic. They pointed out that the environment shapes architects but entrepreneurs shape the environment. The architect builds things intended to work in the world, and hence must take the environment into account. Further, the security entrepreneur was independent, whereas an architect typically worked for someone and hence was dependent on that person or corporation. An entrepreneur in the audience pointed out that it was easier to have a boss to handle much of the business process work (such as laws, patents, etc.). But the security entrepreneur is free to make her own decisions.

Someone suggested that those who were risk-averse were architects, and those who were not were entrepreneurs. Faily replied that the situation was not that clear-cut; everyone has some aversion to risk. The question was one of risk management. Someone else observed that the difference between innovation and entrepreneurship was that "research is transferring money into knowledge, and entrepreneurship is transferring knowledge into money." Also, the diffusion of the innovation was critical to the success of the entrepreneur. Someone noted that the traits normally deemed good were attributed to entrepreneurs, and ones normally deemed bad were attributed to architects. Faily said that these were simply observations, and the researchers took no position on whether the traits were good or bad; they simply were observed. In response, another participant noted that architects have done things to change the world by building well-built projects. The rejoinder, from yet another person, was that architects don't cause anything to be built; entrepreneurs pull together the strands that enable the architects to build. Further, architecture is easy enough so that architects are unnecessary. A large portion of the audience loudly disagreed.

Faily also elaborated on the art of chindogu to prototype security controls. Essentially, chindogu is the invention of a gadget that solves a problem but introduces so many new problems that it has no utility; the example they used was a baby mop (see the paper for the picture). The security example was a "forget-me-not" digital certificate, stored on a dongle that is attached to the picture of a loved one. The theory is that one would not lose the picture and, hence, the dongle containing the certificate. The authors argued that the chindogu can be used to bridge the gap between open innovation (in which ideas and paths to market are generated) and security design.

As a result, predicting the impact of a new security control from different perspectives must be combined with creativity in order to implement innovative security controls. Further, theories from entrepreneurship can apply to security innovation with minimal changes.

- ***Would a 'Cyber Warrior' Protect Us? Exploring Trade-Offs Between Attack and Defense of Information Systems***
  Tyler Moore, Allan Friedman, and Ariel Procaccia

Tyler Moore applied game theory to a simplified version of the computer security problem. The US Cyber Command has among its missions the defense of US cyberspace while exploiting vulnerabilities of its adversaries. This means that the same actors are both attackers and defenders. The bases of the games used were twofold. First, they assume a zero-sum two-player game in which each player has incomplete information. Second, they assume that making vulnerability information public helps everyone (in this context, both players) to defend their systems, and hiding (or "stockpiling") vulnerability information helps only the stockpiling player attack the opponent. They developed two games.

The Simplified Stockpile Game examines the trade-off between stockpiling and defending (protecting society by fixing vulnerabilities). It has player 1 choosing to stockpile only. It has two parameters, one that represents the player's relative technical ability and the other, the social harm of undisclosed vulnerabilities. When the social cost is 0, both players will stockpile. As the social cost increases, the less technically sophisticated player will compensate for their lack by defending (making information public). The Cyber Hawk Game changes the social harm parameter to be one of aggressiveness, specifically the probability that a player will attack after discovering a new vulnerability. Hence it focuses on the costs and benefits of being aware of vulnerabilities, rather than the results of a conflict. The results of this game show that if technical sophistication is equal,

both players will attack; if not, the technically dominant one will be more aggressive.

The discussion focused on the underlying assumptions of the game. First, the question of what a "vulnerability" was engendered considerable debate, ending when the presenter said the definition was axiomatic, but the vulnerabilities of interest had to be shared by the players and be able to be exploited by at least one. It was also noted that the assumption of disclosure of vulnerability information and patches benefiting the community was questionable because not everyone could fix their system before being attacked; this was a simplifying assumption. Second was the use of a zero-sum game to model this situation. A zero-sum game is one in which losses and gains are balanced, and it is not clear that holds here. For example, in a military situation the military is also concerned about budget and other constraints not dealt with on the battlefield. The third, and most important, challenge was the need to add other players (at least one) and consider non-attribution and asymmetry. Some players may have excellent attack infrastructures, but little infrastructure to defend. A third party may attack one player and make the attack appear to come from the other player, causing the two non-aggressive parties to fight. The concern expressed was that the game might create stability points that do not hold in real life, leading to policymakers making decisions based on a simplified model.

The authors intend to expand this model to make it better match real situations. The first step is probably to increase the number of players. The game is nevertheless useful in trying to understand the trade-offs between attacking and defending.

- ***On-line Privacy and Consent: A Dialogue, Not a Monologue***
  Lizzie Coles-Kemp and Elahe Kani-Zbihi

Lizzie Coles-Kemp explained that their goal was to better understand the types of online privacy dialogues that service providers and users want, in order to understand how their data will be used and shared. The study synthesized two layers. Privacy negotiation is supported as part of the physical and information-processing layers, and how it works is fixed because the system providing the negotiation is understood. But at the communal, cognitive layer, the negotiation is treated as a "black box" as part of the process of managing the user, and how it works is variable because we cannot predict human behavior. For such an open system, we cannot determine a full set of variables to constrain, so we can at best postulate partially grounded theories, and not predict behavior. So the question is, what scientific routes are open to the researchers?

Coles-Kemp described three types of dialogue systems. The first provides information about the service provider, users, and peer groups. The second raises issues about how the service provider interacts with third parties. The third deals only with service provider behavior. A particularly interesting finding was that none of the service providers felt they should negotiate the level of privacy they provided with users; they felt this is done through the legislative process. Other findings were that current privacy dialogue systems performed poorly in terms of providing the users with information, because the users avoided reading privacy and user agreements. At the design level, the dialogues need to address how the users can communicate with the service provider and give feedback for improving the dialogue. At the social level, the dialogues must address the conditions and methods under which the provider will disclose information, taking into account the privacy rules from the law, culture, and commonly held beliefs and norms, and must provide sufficient transparency that the user understands how all this will work.

One participant asked whether the services being studied were private or governmental. The presenter said they were communal services, covering everything from garbage collection to land planning to supervision of social work and provision of payments. The key point is that the services were all about relationships and trust with key workers. As these services move online, those relationships and trust change. Someone else mentioned that one can trade privacy for both benefits and trust, and asked if they had looked at the connection between privacy and trust in depth. The response was that inculcation of trust was a building block, and they looked at how that makes citizens less vulnerable. Going online changes the dynamic from that involved in face-to-face meetings, and the issue is how to compensate for the changes in the dynamic. A third question was the truth of the provided information; several people said that if they did not understand why the online provider needed information, they would simply make something up. The presenter said this phenomenon was quite common among younger people they had surveyed, but others called and asked why the information was needed, or simply disengaged. Finally, there was some dispute about whether privacy notices were a true "dialogue," because they typically specify the terms and the user either accepts them and gets the services, or declines them and does not get the services.

- ***A Risk Management Process for Consumers: The Next Step in Information Security***
  André van Cleeff

The premise of André van Cleeff's paper is that users should have a personal risk management strategy for protecting their privacy online. The paper, however, discusses the complexity that arises when attempting to realize such a system in a tool for end users. As one attendee noted, social disclosure of private information also has benefits that such a tool would need to account for. One major issue that the system faces is how to enumerate the different categories of risk, describe risk details, and assign probabilities of events. Another observed that systems like some online health information systems had to have the number of privacy controls reduced simply because user studies indicated that

people became overwhelmed with controls. One benefit of this paper is to help refute the illusion that someone else is doing risk management for you online; as one attendee pointed out, assuming that delegation of risk management works is a moral hazard.

Attendees vigorously discussed whether humans were good at performing risk management at all, but largely concluded that humans *must* do this because no one else is available to do it for us. Attendees noted that the large literature on risk management suggests that no matter how much additional information people see, they still make poor decisions, and that people misperceive risk, particularly in online scenarios. One attendee noted that "risk management" is an overloaded and abused term and suggested a clever alternative definition: "risk management is planning for your next defeat." Another attendee noted that, as in the theme of the paper, if you personalize risk management, it becomes "fear management."

- ***Ontological Semantic Technology for Detecting Insider Threat and Social Engineering***
  Victor Raskin, Julia M. Taylor, and Christian F. Hempelmann

Julia Taylor noted that people sometimes give off a signal by the way they say things or the information they omit in descriptions of otherwise normal events. Casual conversation or conversation in the same context (e.g., with a colleague) often results in relaxed conversation, and people insert or describe new information or modify existing defaults. Another example occurs with social engineering: amateurs often attempt to mimic all or most of the domain defaults in an attempt to establish credibility—words and actions that a real expert would never do. The paper makes the central observation that such deviations from normal speaking or writing patterns convey information that may be of use in detecting certain types of insider and social engineering attacks. Taylor reported on the use of a mature ontology-based technology for understanding the relative semantics of pieces of a sentence (i.e., the main task is not in parsing the sentence itself). Questions by the audience helped clarify that one potential application of such analysis was detection of insider threats, particularly to support further research and analysis once an initial suspicion is formed (and the analyst thereby has legal access to spoken and text corpora of the research subject).

The subsequent conversation focused on clarifying the value proposition of the system. Taylor stated that the system as such was already a mature technology and the product of more than a decade of research. As Taylor went on to say, the modality of the talk is that they have the resources to interpret sentences, but the paper was about additional capabilities to understand what is *unsaid*. Taylor noted that people unintentionally say things that reveal their habits, values, and defaults. Unless subjects are paying attention, they do not notice such disclosure: the point of the system is to do this. Taylor pointed out, in response to a question about what test data set the authors were using, that the point of the paper was to establish the model and theory, and that rigorously verifying the results was a challenging task. As a result, the system is most applicable in scenarios where the corpus is derived from an ongoing session of legal wiretapping or observation of the suspect's communications, although the authors specifically disclaimed any intent to use the system for providing digital evidence strong enough to stand up in court. Several attendees raised concerns about the eventual use of such evidence in a courtroom.

One attendee noted that an interesting source of data to analyze might be derived from an FBI operation set up to infiltrate a specific drug market with online communications; from the perspective of the drug market, the FBI agents were insider threats.

- ***The Pervasive Trust Foundation for Security in Next Generation Networks***
  Leszek Lilien, Adawia Al-Alawnch, and Lotfi Ben Othmane

Lilien began by noting that trust is a pervasive concept in social interactions. This paper examines the logical foundations for incorporating measurements of trust in new computing systems and presents a case study suggesting how one might accomplish this in the context of designing the next-generation Internet. The paper suggested the design of a "pervasive trust foundation," or PTF. Noting that there are degrees of trust and that trust is usually asymmetric and bi-directional (although one direction might be implicit), the paper suggested modeling such relationships to provide security. One major contribution of the paper is to describe how various security services (SS) might be supported or constructed through the use of an underlying "Trust in the Large" (TIL) subsystem. The paper then considers obstacles to finding an efficient PTF/TIL implementation and suggests approaches for decreasing the performance burden of checking security properties by taking advantage of the TIL layer.

Discussion included a wide variety of interesting issues, as trust seems to be a cross-cutting issue in the information security field. One attendee pointed out that the definition of trust offered in the paper might need to be augmented with the concept of trust sourcing: building trust requires a well-founded evidentiary process to establish a trust basis. One major theme of the discussion was how usability of a software artifact leads to a simple mental model, which then leads to or creates trust in the operation of the software artifact. It was noted that such a "contract" is a substitute for security in the sense of measuring real properties and constraints on execution. The discussion also focused on how security typically implies some objectively measurable value; trust is usually a subjective notion—in many situations, appropriate security measures should be determined by the size of the threat, not the level of confidence users have. Finally, one interesting technical issue that was raised was how to deal with efficiently revoking trust relationships (given the assumption of a pervasive trust foundation).

- ***This Is Your Data on Drugs: Lessons Computer Security Can Learn from the Drug War***
  David Molnar, Serge Egelman, and Nicolas Christin

David Molnar proposed using observational data to draw conclusions about computer security. The controversial issue is what data is useful, what data is biased, and how we compensate for that bias in order to we might draw sound conclusions from the data. The authors draw a parallel between the observational data accumulated over the years in the drug wars; there is a body of data (the STRIDE data, a time series of price and purity of the drugs seized) from drug busts, and this data is used for research. It is controversial, in that the data was not intended for use in research, and so the biases inherent in it must be noted and compensated for. Further, the data comes from those who have been caught and does not reflect the price differences between those and others who do not get caught. The authors then ask what we can learn from the drug wars. Their hypothesis is that applying lessons from the analysis of the STRIDE (and other) data will improve observational research in computer security.

Molnar noted that papers drawing conclusions from observational data in computer security have a shaky basis; indeed, one well-known paper on purchasing services to crack CAPTCHAs was known to be flawed because it drew unwarranted conclusions. A participant asked if this demonstrated naïveté on the part of the authors or reviewers or both, and the presenter responded that that is the reason he brought up the controversy about the STRIDE data. A second participant said that the authors of the CAPTCHA paper probably felt the data, although biased, was better than no data; the retort, from another member of the audience, was that it would be worse, because if you are using data you know to be flawed, you must reveal the flaws and the possible consequences of using that data. This led to a discussion of what kind of conclusions the experimenter can draw, and the consensus (such as it ever is at NSPW) was that the authors must be clear about the methods they used and demonstrate that the conclusions are reasonable given the data (biases and all). It was also pointed out that errors may be caused by unknown factors as well as malevolence and incompetence; for example, when measuring the gravitational constant, a series of experiments used different methods, and lots of the research was to analyze others' methodologies to find out why the results disagreed. This is actually supportive of the experimenters, because it helps them refine both their methodology and their understanding of the experimental process. It reveals things that were not known before.

The discussion then moved back to the analogy of data from drug busts and data from computer security. In their talk, the presenters had observed a huge price dispersion in drug prices (the figure cited was a $6000 difference in the price of a kilogram of cocaine in New York and in Boston). Someone noted that the price dispersion for drugs was based on physical separation and asked if this was also true for computer security data such as stolen credit cards or attacking CAPTCHAs. The presenter pointed out that the problem was not physical separation but logical separation, specifically the different groups with different rules of access creating different markets—and it was necessary to study these differences in order to understand how they affected price. The discussion concluded with some thoughts on experimenter bias, the well-known ways other fields compensate for that, and the question of how to do the same in computer security experiments.

- ***Relationships and Data Sanitization: A Study in Scarlet***
  Matt Bishop, Justin Cummins, Sean Peisert, Anhad Singh, Bhume Bhumiratana, Deborah Agarwal, Deborah Frincke, and Michael Hogarth

Matt Bishop focused on an interesting path in data sanitization. Although the field has had a lot of examination (from privacy-preserving databases to sanitizing network traces), the authors have been examining a new paradigm for dealing with a major challenge in this space: the availability of external knowledge to an attacker wishing to reverse the sanitization effect or otherwise infer some knowledge from the sanitized data trace. One key insight is that the authors treat sanitization as a problem of *risk*, not certainty. They assume that (1) relationships used by attackers are unknown to the sanitizer, (2) effective sanitization might not exist, and (3) inferences might not be correct, but incorrect conclusions are potentially damaging. As a consequence, they construct a framework for asking, "What relationships enable desanitization?" and "How likely is it that the existence of these relationships is discoverable?"

The framework Bishop presented uses an ontology to help reveal the conflicts between a privacy (or sanitization) policy and an analysis policy. The threat model informs the structure and content of the privacy policy, and security requirements (or whatever domain requirements exist for the specific analysis being performed) inform the structure and content of the analysis policy. The system logically performs analysis on both the raw data set and a sanitized version of that data set. The system compares the results and produces a set of conflicts arising from the two policies. The system enables an expert to help resolve these conflicts. The overall purpose of the work is to help inform users and organizations interested in data sharing about the risks specific to their activity. Bishop also observed that one big practical problem is how to create a "consumer-friendly" assurance argument.

The discussion largely focused on attempts by the audience to understand the semantics of sharing, (de)sanitization, and policy construction. The audience was also curious about the role of the ontology. Bishop asserted that the ontology was useful in helping reconcile the fields in the privacy and analysis policies (as these may come from different domains and use different terms and descriptions). Bishop also noted that one big win with using an ontol-

ogy over a simpler mapping of field relationships was that several tools already exist for manipulating ontologies that make such comparisons much easier than writing a system from scratch. Attendees noted references for several types of desanitization attacks, including inserting marker data and whether it was a risk that an attacker might reverse-engineer the ontology. One observer noted that this work would be valuable because many times in research, a large data collection and sanitization effort is undertaken that is wasted when researchers realize that they can't use or analyze this data in a meaningful way.

PANEL

- ***Why Is There No Science in Cyber Science?***
  Panelists: Richard Ford, Carrie Gates, Lizzie Coles-Kemp

The last presentation was a continuation of the opening panel, but with a surprise: three new panelists presented their thoughts on cyber science.

Richard Ford began by arguing that science is a friend, not an enemy; it is how we actually produce knowledge. Change is incremental and slow, but can begin now—for example, by rejecting papers that do not demonstrate good science. Doing better science does not mean that we will be any less productive; while it is harder, the results are much more long-lived than non-scientific results. So, we must ask ourselves: do we really want to understand our world, or just get published?

Lizzie Coles-Kemp followed. In every artifact, there is a physical object and a social object, and we need to respect this duality. How we produce knowledge about each dimension has its traditions. Some papers presented over the past three days straddled the physical and social worlds. What are good scientific methods in each of these paradigms? How might we use them in each, and how might we take this forward?

Carrie Gates went last. She argued that there was an industry perspective involved. Research should make a difference; indeed, only useful research is good, whether or not it is scientific. Time is of the essence because if the work takes too long, a competitor will grab market share and the company waiting for the science, or for the results, to be scientifically valid will lose; in other words, science takes too long. Incremental improvements are good enough; indeed, the perception of improvement is good enough, even if in reality there was no improvement.

Almost everyone in the workshop indicated that they wished to speak. Someone pointed out antivirus as an example of the need for non-incremental research. Current antivirus software is very poor, due to the near-term focus and incremental approach used to improve the software. If we applied more science to it, we might obtain better results. Gates asked why researchers had not given industry something better than the current mechanisms, and the response was that better ways were known, but the market has yet to adopt them. Someone else suggested that one needs science to produce generalizable results; the response was that one should not conflate product development with fundamental research. Another comment was that industry might not want scientific results that they can use; they focus primarily on whether users need their product, not on whether the product does exactly what the sales force claims it does. The question of how to determine whether something is useful arose, with a participant noting that what may seem utterly useless (the example used was Reimannian geometry) may turn out to be extremely useful (in the example, it was found to be the actual geometry of Einstein's theory of space-time). The panel agreed with this point, arguing that the requirement that results be useful immediately is killing the field.

Someone asked for ideas on how we might change the culture of computer security to be more scientific. Suggestions included not rejecting papers that had claims not supported by science, but instead working with authors (possibly through a shepherding process) to ensure that the claims are appropriate to the work done; requiring a methodology section describing how the experimental work was done; and releasing code and data whenever feasible. With respect to this last point, someone else said that when the research involved the use of proprietary code, releasing the code may not be possible and so if code cannot be released, the results should not be automatically rejected. The panel reminded everyone that as reviewers, we have considerable power to change the culture, and we should use it.

Each panelist then said a few words. Gates argued that the best way to get industry to value and use research is to embed researchers in the different industries: this would communicate the research results in a way that could be incorporated into products. Ford commented that we should change the culture in small steps, and think about how best to communicate the needed changes to others. Coles-Kemp concluded that more venues such as this workshop would raise awareness of the problem and ways to change the culture.

NEXT WORKSHOP

The next NSPW workshop will be held at the Marconi Conference Center in Marin County, California, from the evening of September 12, 2011, through lunch on September 15, 2011. Sean Peisert will be the general chair and Richard Ford will serve as vice-chair. Carrie Gates and Cormac Herlihy will lead the program committee. Submissions will be due by April 4, 2011. Details on how to submit papers will be posted to the Web site http://www.nspw.org in the near future.

VMware Sponsored Academic Research Awards – Request for Proposals (RFP)

Theme: Performance Management Challenges in Virtualized Environments

Virtualized Environments are proliferating in the IT industry. They are becoming a foundation of many computing and communication environments from large enterprises and multi-tenant clouds to virtualized desktops, as well as mobile endpoints. The management of performance in such virtualized environments provides many interesting challenges.

In this RFP, we are requesting proposals that cover but are not limited to any of the following areas:

- Development of large-scale statistics gathering and analysis in scalable virtualized environments. Specific areas of interest include but are not limited to health models for multi-tier applications as well as correlations application, VM and host performance.
- Improvements in coordinated resource management across applications, VMs and hosts. For example, transparent solutions for the double-swapping problem and elimination of redundant disk I/O across VM and host, as well as better management of runtime systems such as JVMs and databases in overcommitted situations in VM and host.
- Performance improvements of an emerging class of distributed, latency-sensitive programming and middleware frameworks, such as Hadoop, Memcached, GemFire Data Fabric and traditional HPC. This includes but is not limited to performance studies, performance optimizations, virtualization enhancements, and explorations of novel approaches that increase the value of these new frameworks through integration with virtualization.
- Performant and scalable handling of all virtualized environment management data, including data consistency, data distribution and levels of coupling between management and managed elements. For example, design of a scalable management and monitoring infrastructure for millions of VMs.

Award Information

Funding Award	up to $250,000
Duration of Proposed Research	One to three years (preference for 1 year projects, renewable based on results)
Funding Type	Gift or Grant (to be determined on a case by case basis)
Student Internships	One summer internship for each summer for each student for the duration of the project
Submission and Correspondence	*funding@vmware.com*
Program URL	http://vmware.com/go/vmap-rfp

Proposal Evaluation

Phase 1

Preliminary Proposal Page Length	One page short technical proposal with coarse grain deliverables
Preliminary Proposal Due Date	January 10, 2011
Preliminary Proposal Notification	February 4, 2011

Phase 2

***Optional* Exploratory Virtual Workshop**	Principal Investigators chosen for Phase 2 are invited to an optional ½ day virtual workshop on Performance Management Challenges in Virtualized Environments
Workshop Date	February 18, 2011
Full Proposal Page Length	10-20 pages
Full Proposal Content	Technical approach, preliminary results, deliverables, milestones, budget, CVs
Full Proposal Deadline	April 8, 2011
Award Results Announcement	April 29, 2011
Project Start	May 27, 2011

vmware

USENIX
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