usenix ;login:
DECEMBER 2012 VOL. 37, NO. 6
ELECTRONIC SUPPLEMENT

Conference Reports

21st USENIX Security Symposium (USENIX Security '12)

Bellevue, WA
August 8–10, 2012

Opening Remarks

Summarized by Rik Farrow (rik@usenix.org)

Program Chair Tadayoshi Kohno (University of Washington) opened the conference by telling us that there were 222 papers submitted and 42 accepted. By replacing four invited talks sessions with paper presentations, more papers could be accepted than in the past.

When the conference began, there were 484 attendees; 84 students received travel grants using money provided by the National Science Foundation, with Google and Microsoft being the next largest sponsors.

Best Paper awards went to "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices," Nadia Heninger et al., and to "Social Networking with Frientegrity: Privacy and Integrity with an Untrusted Provider" by Ariel J. Feldman et al. (Best Student Paper).

Keynote Address

Summarized by Sarah Meiklejohn (smeiklej@cs.ucsd.edu)

The Evolution of Information Assurance

Dickie George, Johns Hopkins Applied Physics Laboratory

As early as the ancient Greeks, individuals and nations have been interested in protecting information and in providing information assurance, i.e., some guarantees about the secrecy of information. Historically, one area in which information assurance has been hugely important is communications, in particular within the context of a war. In World War II, the benefits of securing communication (or rather, the shortcomings of not securing communication) became apparent in the context of Enigma, the German cipher. As a result of a weakness in the Enigma cipher, a combined effort by the Americans and British allowed them to decrypt ciphertexts encrypted using Enigma; this effort was not included in the German's potential "threat model," as they considered the attack too expensive to be feasible.

After WWII, the mentality therefore changed. During the Cold War, American cryptographers began to consider formal adversarial threat models (i.e., what the adversary could and would do). This was characterized along a number of different vectors: resources, capabilities, and access (what they could do), as well as intent, motivation, and risk aversion (what they would do). The US knew that the USSR had plenty of money, scientists, and technology (resources), and that they also knew what the US was doing (capabilities) through the use of spies in the US and their interception of our ciphers (access). After considering all of these different criteria, cryptographers could then attempt to provide information assurance in the face of this particular enemy. Again focusing on communications, they examined the security of the SAVILLE cipher and its usage in the VINSON hand-held radio. Additionally, they really had time to do it right: the SAVILLE cipher was developed in the 1950s, evaluated in the 1960s, and then finally implemented in VINSON and deployed in 1976; this meant that they could look closely at both the algorithm and its implementation to try to find potential attacks.

In modern times, the field has once again changed dramatically. One of the main catalysts for these changes was the introduction of the Data Encryption Standard (DES), which was created through a competition held by NIST. The eventual winner of the competition was the Lucifer algorithm developed by IBM; this original algorithm was then tweaked by the NSA, which changed the S-boxes to prevent against an attack using differential cryptanalysis, which at the time they were aware of but the public was not. The big change initiated by this competition was the introduction of the public: ordinary people were being encouraged to evaluate DES, commercial uses for encryption were being developed, so it was no longer just nation states using it; the algorithms were becoming more complicated at the same time that users were becoming less expert. George provided the example of one company where users "didn't need key management" because they simply used the key the system came with. Going hand in hand with the development of commercial products, cryptographers no longer have the luxury of 20 years to look for security flaws, which means these flaws are discovered only after the product is already deployed.

Nowadays, things are far less simple than just saying that "Russia is the adversary." The rules have changed and, as exemplified by Stuxnet and other cyberattacks, there is no longer the same sense of "honor amongst spies." Additionally, the entire threat model has changed: to launch a widely destructive attack, one need only have access to an Internet-connected laptop. Furthermore, the symmetry we had during the Cold War is gone: an attacker now is not worried about who strikes first, or retaliation, and we fear individuals more than nation-states. China or Russia wouldn't be willing to take down our power supply or financial system, as they are too connected to us, but some lone terrorist would. Finally, the targets of these attacks also have shifted from governments to ordinary citizens, who are now making the same

risk management decisions as the NSA was, but without the background to solve these problems themselves.

Many questions from the audience were about either DES or Stuxnet. For the former, Ian Goldberg (Waterloo) asked why the key size was changed to 56 bits. George responded that the NSA had an agreement with NIST not to "overpromise" security, and that 56 bits would provide security against their threat model at the time until 1990. Matt Blaze asked why the NSA modified the S-boxes knowing that it would inform the world about differential cryptanalysis or even why they participated in the first place. George responded that it was a risk management decision, but that ultimately the NSA cryptanalysts were best capable to evaluate the algorithm at the time, and that it made the NSA a more open agency as a result. Perry Metzger asked whether they also were aware of linear cryptanalysis at the time. George responded that they didn't understand it as well, and that back then they also didn't believe that DES would be used with enough text to launch a linear cryptanalysis attack.

Doug Roberts asked whether Stuxnet was sophisticated enough that it must have been written by at least one state. George said that it was indeed very carefully crafted and in all likelihood was written by a nation-state, but that he has intentionally avoided reading any classified reports so couldn't say for certain. Stefan Savage then asked if any of the "honor amongst spies" was left or if Stuxnet had demonstrated that it wasn't. George replied that there's always a technology "creep" and the line in the sand is certainly drifting over. Juan Lang asked what he thought the trend was now that Stuxnet demonstrated that espionage might not even be between states. George responded that he doesn't envision a rosy future, as the exposure of Stuxnet has given every potential terrorist a tool to analyze and work with. Steve Bellovin asked about the evolution of worms such as Stuxnet and also Flame, which used cryptanalytic techniques "unknown in the open literature." George responded that he wasn't sure the techniques really weren't known, but that nation-states should be able to figure things out that the public doesn't; wouldn't it be embarrassing if they didn't?

Yossi Oren asked what the right approach was to take when finding a vulnerability, as many companies were reluctant to update their products. George responded that it was always much easier for the NSA to get listened to, that threatening to go public didn't seem like a great solution, but that he agreed it was very important to get things fixed. Finally, Zach Tatlock asked whether he saw formal verification, in which you prove that implementations do what they are supposed to, as a potential solution. George said that in theory it was, but that they would first have to make it more scalable.

Spam and Drugs

Summarized by Amir Rahmati (rahmati@cs.umass.edu)

PharmaLeaks: Understanding the Business of Online Pharmaceutical Affiliate Programs

Damon McCoy, George Mason University; Andreas Pitsillidis and Grant Jordan, University of California, San Diego; Nicholas Weaver and Christian Kreibich, University of California, San Diego, and International Computer Science Institute; Brian Krebs, KrebsOnSecurity.com; Geoffrey M. Voelker, Stefan Savage, and Kirill Levchenko, University of California, San Diego

Damon McCoy discussed the shape of the underlying business enterprise of online pharmaceutical companies, or as he called it, "rogue pharmacy economics 101." There are three main players in this economy: user, affiliate marketers, and affiliate programs. In this study they looked at the relationship between these parties by analyzing the ground truth data of more than $170 million money transaction data leaked from GlavMed, SpamIt, and RX-Promotion, along with order information and the chatlog of GlavMed/SpamIt.

One thing to remember about the affiliate programs is that they are here for the long haul. Their customers usually use credit cards for payment and, as such, they need to have satisfied customers in order to maintain their business. They also need to maintain good relation with their affiliates, suppliers, and payment processors. In this study, the researchers looked at various characteristics of customers and affiliates such as demographics and shopping patterns, new customer streams, and affiliate revenue and commissions.

Using these data the authors have concluded that only a small number of advertisers and payment processors are responsible for the majority of sales in this economy, and by hindering the top payment processors, the authorities can hugely affect the success of these companies. Furthermore, their studies have suggested that the customer base of these companies is steadily increasing without any sign of slowing down. Although many people purchase "franchises" from the affiliate marketers, very few actually succeed, that is, make even $5000 before going out of business.

Adam Langley (Google) asked why Bitcoin was not used. McCoy replied that the customers seem to prefer credit cards and 95% of transactions are done using that. David Wagner (UC Berkeley) asked whether these data can guide any policy-making decisions on drug controls. McCoy answered that they aren't able to make any connections.

B@bel: Leveraging Email Delivery for Spam Mitigation

Gianluca Stringhini and Manuel Egele, University of California, Santa Barbara; Apostolis Zarras and Thorsten Holz, Ruhr-University Bochum; Christopher Kruegel and Giovanni Vigna, University of California, Santa Barbara

Gianluca Stringhini presented a new methodology for spam detection. Traditionally, spam detection has been done by

either content analysis or spam origin. Their work focuses on the email delivery mechanism, and analyzing the communication at the SMTP protocol level. They introduced two techniques for spam detection.

The first method, called SMTP dialect, follows the finite state machine of spammers' SMTP protocol, which has been tweaked to achieve high send-out speed and use passive matching and/or active probing to detect and stop them. Passive matching in this system would follow a normal SMTP protocol and try to understand the dialect from the normal transaction. In active probing, the server will issue error messages that will try to distinguish a spammer from a valid email sender.

The second method, called feedback manipulation, uses SMTP feedback to create a lose-lose situation for the bot master. The system will send an email "recipient does not exist" message back if it detects that an email is spam. The spammer then has the choice of removing the email from the list to increase the performance or ignoring it and accept the overhead. Data shows that about 35% of email addresses are dead addresses so not deleting the non-existing addresses will induce a large overhead on the system.

Kevin Fu (U Mass) asked whether they also have considered timing responses in recognizing dialects. Gianluca said that they haven't had access to timing data but such data can definitely be used. Jeremy Epstein (NSF) asked whether they considered user privacy issues in their work. Gianluca replied that user data was anonymized.

Impact of Spam Exposure on User Engagement

Anirban Dasgupta, Yahoo! Labs; Kunal Punera, RelateIQ Inc.; Justin M. Rao, Microsoft Research; Xuanhui Wang, Facebook

The first question Justin Rao asked was, "Is spam bad?" The answer to this question depends on whom you ask, but a study has shown that the social cost of spam compared to any monetary return is 100:1. That this ratio is 19:1 for car theft and 0.04–0.37:1 for driving a car shows how deficient the spamming process is.

So the question for Yahoo as a mail provider is how much they should spend to fight spam. To answer this question, the researchers tried to gauge user dislike of spam by measuring the causal effect of spam exposure on user engagements and how it evolves over time. Two main challenges existed for this study: an A/B test was not feasible and spam exposure is endogenous.

To get around this, Rao and his team developed a large-scale nearest-neighbor matching technique. Using this method they found the nearest neighbor of each user by matching various factors for two months. The average result they got is what you would expect: user engagement decreases with higher exposure to spam. An interesting result in this research was that volitional user actions such as composing, replying, and forwarding get affected more than responsive actions such as reading. Additionally, exposure to spam has a cumulative negative effect on engagement. The tipping point for how much spam is too much for the user is about 1 spam per login.

Kevin Fu (U Mass) asked how much spam protection is enough. Rao answered that it really comes back to how much better than your competitors you have to be for the users to care, and the answer isn't an easy one. Chris Kanich (U Illinois, Chicago) asked how many users will create enough activity to have statistically significant conclusions from the data. Rao answered that the pilot sample was 2 million people and they got tight matches through 35,000 pairs, but the statistical significance wasn't "something to act on," so they analyzed all their data. Someone asked whether other providers' performance on spam filtering affected this study. Rao answered that it would have but because the study wasn't done over a long time span, it was their hope that users' options remained constant throughout the study. Someone else asked whether they considered using Yahoo Mail's own advertisements in their studies and whether ads might have affected the research because those who couldn't tolerate spam had already fled. The use of Yahoo's own advertisements was not viable because the ads were expensive and their use would have negatively affected their relationship with Yahoo. Rao admitted that there can be some selection effect because of the ads, but they had no way of considering those.

Invited Talk

Summarized by Alexandru Totolici (totolici@cs.ubc.ca)

Robots and Privacy

Ryan Calo, Assistant Professor, University of Washington School of Law, and Affiliate Scholar, Stanford Law School Center for Internet and Society

Ryan Calo provided an overview of the privacy concerns surrounding the increased levels of interest and availability of robots, addressing primarily matters concerning governmental use. Costs have been constantly trending downwards and, therefore, robotics are predicted to take off dramatically in both private and public sectors. Roboticists have focused, so far, on getting things working, ignoring most security matters—as has been the case in general-purpose computing for a long time. Calo argues that it is time to get involved and ensure that these new devices are adequately provisioned in terms of security and privacy. For a more in-depth treatment, see Calo's "Robots and Privacy" paper (http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1599189).

Calo began the discussion by settling on definitions for "robots" and "privacy." A robot is any machine that can sense, process, and act accordingly, and these characteristics are what distinguish a drone from a remotely controlled plane with a camera. Privacy, while a very difficult concept to describe and agree upon, is taken here to mean an individual's control over his personal information.

There are three principal ways in which robotics implicates privacy: increased capacity for direct surveillance; access to historically private places and data; and social consequences of robot presence.

The ability to conduct direct surveillance is greatly enhanced by robots due to a combination of low cost, form factors, and advances in computer algorithms able to automate decision-making based on sensory inputs. Federal and local law enforcement have already begun using drones and quadcopters for border surveillance, traffic, and crime scene surveillance. "Avatar" is a virtual border agent in Arizona, trained to detect "honest signals" (subtle cues in a human's response to various questions). Novel robot types are constantly under development, such as the DARPA-funded Hummingbird (an unobtrusive flying drone) or the wall-climbing robots of SRI.

Many of these novel robot types also permit government access to historically private places, either directly or indirectly. For example, gaining access to above-ground windows is significantly easier with drones, enabling dragnet surveillance. Indirect access to collected data is also possible, either through security breaches (sniffing communications between drone and base, or spoofing GPS and capturing the drone itself) or as a result of a search and seizure warrant executed by law enforcement; without adequate protection, a home robot may be a treasure-trove of personal information.

Lastly, there are a number of social consequences associated with robotics. Many robots are given anthropomorphic features to improve their interactions with the elderly or the disabled they help care for, as humans are hardwired to respond to such cues. The downside, however, is that most users may find themselves less forthcoming when interacting with such interfaces. One example is that of Microsoft's "Ms. Dewey" search engine, and the question of how search terms may change if the popular engines were using similar avatars.

As an overall theme, drones commoditize large-scale surveillance to a degree the current legal framework is not equipped to handle. In the absence of additional legal and technological factors, the proliferation of personal robotics also exacerbates the risk for indirect privacy violations. The multitude of positive applications of robotics cannot be ignored, thus the need for adequate protections is an urgent one.

CAPTCHAs and Password Strength

Summarized by Benjamin Mood (bmood@cs.uoregon.edu)

Security and Usability Challenges of Moving-Object CAPTCHAs: Decoding Codewords in Motion

Y. Xu, University of North Carolina at Chapel Hill; G. Reynaga and S. Chiasson, Carleton University; J.-M. Frahm and F. Monrose, University of North Carolina at Chapel Hill; P. van Oorschot, Carleton University

Yi Xu assessed moving image object recognition (MIOR) CAPTCHAs in terms of security and usability, listed an attack, and suggested an improvement. In a MIOR CAPTCHA, the user has to type the specified letters that are moving and slightly rotating on the screen. The advantage for the user is it is easy to use; however, there are multiple disadvantages that an attacker can use. For example, MIOR provides multiple views containing temporal information, which can be used to enhance attacks, and it relies on cognitive tasks instead of object classification. Xu noted the latter is hard for computers.

Xu presented both a naïve way and a cutting-edge way to break MIOR CAPTCHAs. The naïve way is to pick a frame, extract the letters, and attempt to determine the letters. They presented a new algorithm that tracks each individual object across frames, extracts the letters, splits the string into different segments, and then uses a neural network to determine each letter. Once a character is known, it can be removed from the image and then the process can be run again on the image where the solved letter has already been removed. This last step is referred to as feedback. They tested 200 MIORs across 19 backgrounds. The results presented are as follows: the naïve method could distinguish single characters 65.5% of the time and three characters 36.3% of the time. Their algorithm without feedback worked 90.0% for single characters and 75.5% for three characters. With the feedback loop, the algorithm worked 90.3% of the time for single characters and 77.0% for three characters.

To improve MIOR CAPTCHAs, they suggest using "Emerging Images." This adds noise, which prevents attacks from working correctly. In an emerging image, each frame is almost completely different from the next. They tested multiple varieties of MIOR CAPTCHAs (increased word length, overlapped letters, and semi-transparent text) and compared it with Emerging Images. They found Emerging Images was the only MIOR that stopped their algorithm. Current MIOR CAPTCHAs do not offer sufficient protection. Emerging Images is a good approach based on today's attacks. A better approach to CAPTCHAs might be classification of identification of high-level semantics.

Jeremy Epstein asked what is the computational cost, as we can solve CAPTCHAs for a fraction of a penny online? Does this change the human cost? Xu responded that their

aim was to find the best defense against computers; using humans is beyond them. Epstein then asked how much computation a solution takes. Xu said that it takes one minute to solve a video-based CAPTCHA. But they did not try to make it fast (they used a meta-code language and not C or parallelized computation); it is still comparable to using human resources. Someone else asked whether there is a possible way to expand CAPTCHAs by identifying which letters to type. Xu said that each letter should keep its own trajectory. Even if more letters were jumping, as long as it is not random, humans can still figure out which is which. Humans also need the letters to maintain the same trajectory. Another person asked why feedback did not greatly improve accuracy. Xu responded that their experiments only had three different letters, and improvement is not very noticeable with a small number of letters. With the feedback loop, the accuracy solving gets boosted for CAPTCHAs with a larger number of letters.

How Does Your Password Measure Up? The Effect of Strength Meters on Password Creation

Blase Ur, Patrick Gage Kelley, Saranga Komanduri, Joel Lee, Michael Maass, Michelle L. Mazurek, Timothy Passaro, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor, Carnegie Mellon University

Blase Ur presented this study on password meters. His team wanted to answer the following questions: how do password meters affect passwords and user sentiment, and what elements of meter design are important? They performed a 2,931 participant online study using a "between subjects" design. They had 15 possible tests; it was a two-part study over two days in which they had people create a password and then answer survey questions about it. The only requirement was that the password have eight or more characters. They also had the user reenter the password eight hours later and answer a survey about how they remembered the password or how they didn't remember the password. Participants were found via Amazon's Mechanical Turk, and were 63% male and 40% technical. They measured composition, guessability, creation process, memorization, and sentiment.

There were two control conditions: no meter and a baseline meter in which eight lowercase letters filled one-third of the meter. Higher scores were achieved by longer passwords or using different character classes. The baseline meter was not segmented, there was a word corresponding to how much of the bar was filled ("Fair," "Good," ...) and a suggestion for how to make the password better. They also showed the user whether the password was in a dictionary. The other meters they used varied from the baseline meter in their visual appearance and how much of the meter was filled (scoring differences). They checked passwords for guess resistance by checking them against most used passwords.

They tested the following visual differences: segment vs. continuous, color change on a better password, size of meter, no suggestions for better passwords, and the dancing bunny meter (bunny dances faster for better passwords). There were four scoring differences: half score of baseline, 1/3 score of baseline meter, push user toward longer (16) characters, and push toward multiple character classes. In their results they found the more stringent meters resulted in passwords that are less likely to be guessed. They found the different meters do not make any difference in whether or not a person could remember their password. Ultimately, they found stringency in the meter is important and having a visual meter is important. They also found size, color, segmentation, and bunny dance speed are less important.

Shane Clark said that he liked the study, liked the large sample size, but wondered whether there was a user tendency to give up on a password. Ur said that this is covered a little in the paper, and that there is a tendency to fill the meter even if not required; however, they did not find that the 1/3 score meter produced the longest passwords so there was some tendency not to fill the meter. With the 1/3 score meter, people did not care whether they had a high score. James Heather said that telling people about dictionary attacks seemed odd. Ur said that they did tell them whether their password was in the OpenWall cracking dictionary in all cases. Serge Egelman (UCB) asked about the ecological validity of the study design. Ur answered that participants knew they were in a password study, but they only had to create an eight-character password and would still get paid the same amount. The ideal situation would be to control a major Web site's password creation.

I Forgot Your Password: Randomness Attacks Against PHP Applications

George Argyros and Aggelos Kiayias, University of Athens

George Argyros presented on attacking randomness in PHP. They attacked randomness during password resetting by trying to predict the token used to reset a password if a user forgets his or her password. This is enabled since the PHP core does not have a cryptographic PRNG. The built-in PHP functions to generate randomness are not secure. Various applications have this vulnerability, including the USENIX submission script. If a hacker has the exact second for the request, a brute force attack will achieve success after 500k tries. They developed two algorithms to help in this attack: "Adversarial time synchronization" synchronizes with server time, and "request twins" approximates the server time. They created a PHP script to predict the random numbers affected by both round trip time and CPU time. In both algorithms, the time taken for a successful guess is reduced from the brute force baseline.

If they can interact with a newly seeded generator, predicting the seed is possible. To do this, they have to generate fresh processors and interact with them for multiple requests. Then they used a side channel leak to get the session identifier. If they can get the session identifier then they can predict the seed. Since entropy is 40 bits long in the session identifier, they can brute force the session identifier in a few minutes for $700 by outsourcing the computation. If the generator is not seeded, then the random seed is calculated by an equation. Because the seed is only 32 bits, it can be brute forced (or a user could use a rainbow table). Since all PRNGs in PHP are linear, after observing outputs they can recover state. They described in detail how to retrieve the state of the random number generators, which involves solving the truncation problem (how to determine the state when part of the number is left off).

PHP 5.4 added extra entropy to the session identifier, but the direct brute force is still feasible. Argyros suggested adding a secure PRNG to the PHP core. When they contacted the PHP team, they were told it is an "application-specific problem" even though it is a problem for all of PHP. Randomness attacks affect a large number of PHP apps. Ways to mitigate these attacks are needed.

Ian Goldberg, Waterloo, asked what advances they had over the "hot or not" paper. Argyros said that he can't tell for sure, as he just found out about it yesterday, but they do something similar; he suggested they take this offline. Perry Metzger (U Penn) said that based on the graphs shown, the algorithm slows down when it calculates more bits in some circumstances, but those equations do not give information. Did they consider precalculating those bits? Argyros said that because of truncation, whether or not a bit is useless is not known, but they probably could have done something about it.

Invited Talk

Summarized by Michael Dietz (mdietz@rice.edu)

Crowdsourcing

Rajesh Patel, Microsoft

Many new technologies and systems are built to take advantage of the cheap, accessible computational power available in the cloud, but Microsoft Senior Program Manager Rajesh Patel argues that the technologies required to harness the untapped people power made available from crowdsourcing have lagged behind. Although there has been some success in harnessing the potential of the "crowd" of distributed Web connected experts and laymen by projects like Amazon's Mechanical Turk, Patel argues that gathering useful data from these tools is an art, which keeps the untapped expertise provided by the crowd out of reach for most individuals and companies.

Patel sets forth a dream that in the future any task that would normally be outsourced could instead be crowdsourced and that a Web-connected expert could be called upon to apply her expertise to solving the problem. The speaker opined that in order to approach this goal there needs to be a platform that encapsulates the current art of crowdsourcing and allows businesses and individuals to tune the assignment of tasks, retrieval of data, and worker compensation in order to match their differing needs.

Patel then discussed the design of an ideal crowdsourcing platform: one that can support a marketplace for task brokers, match tasks with candidates qualified to perform the task, and handle the transfer of funds between brokers and workers. This proposed platform would deal in small atomic task units called "microtasks." These small tasks would allow for timely, repeatable, and high-volume responses from workers that correspond nicely with the needs of a business trying to replace traditional outsourcing with crowdsourcing. Some examples of these microtasks include generating machine-learning training data, language translation, and A/B testing, which are all tasks that require human input and map poorly to the current model of hiring more full-time employees or outsourcing when demand exceeds available productivity.

The majority of Patel's discussion revolved around the challenges that the developers of a crowdsourcing platform must understand and cope with in order to provide value to the platform's customers and workers. These challenges can be divided into the broad categories of planning and quality control.

The initial planning challenges that a crowdsourcing platform must address primarily involve goal setting and output expectation. The producer of a set of tasks must be able to set a high-level goal for the tasks he submits—e.g., "Determine which of eight logo designs did the participants prefer." This high level goal must then be broken down, by the platform or producer, to determine worker-facing output expectations and instructions such as, "View four sets of two logos and pick your favorite from each set." Patel claimed that this initial design and planning phase of crowdsourced tasks is critical in that it determines the eventual parallelizability of the set of tasks.

The quality of results can skew the results of crowdsourced tasks, so a crowdsourcing platform must understand how quality can be degraded, both intentionally and unintentionally, and defend against it. Patel pointed out that in his experiences with crowdsourcing, workers regularly did the minimum required to complete a task, didn't understand the guidelines, shared a single account with others, and used

programs to perform some tasks. All of these issues can affect the quality of an entire result set, potentially invalidating hours of work, so there needs to be some mechanism for validation of results. Patel claimed that in his experience, result validation is very expensive in practice and heavily relies upon customer expectations set in the planning stage of the task life cycle.

Patel concluded by pointing out that although crowdsourcing is not a new idea, the technology required to tap into its potential has not been present until now and that the time is ripe for research into this new area.

An audience member asked about the study of human subjects via crowdsourcing and the role institutional review boards will need to play in governing the ethics of doing research on gathering data on workers in the cloud. Patel responded that this has not been an issue in their research at Microsoft as their legal team fills the role that an IRB would perform at an academic institution.

A second audience member asked about Microsoft's crowdsourcing platform offerings. Patel responded that Microsoft has its own private platform and that it performs about 10 million tasks with it per month.

Browser Security

Summarized by Gianluca Stringhini (gianluca@cs.ucsb.edu)

An Evaluation of the Google Chrome Extension Security Architecture

Nicholas Carlini, Adrienne Porter Felt, and David Wagner, University of California, Berkeley

Adrienne Porter Felt presented a study on the security of the Google Chrome extension architecture. Malicious Web sites might try to exploit extensions to get access to the browser API or other Web sites, or modify benign Web sites served through HTTP. Google Chrome has three security mechanisms in place to prevent this: privilege separation, isolated worlds, and permissions. In particular, extensions are built from two types of components: content scripts, which interact with Web sites with no privileges; and core extensions, which execute the extension with full privileges, but do not interact with Web sites. The question the authors are trying to answer is whether such security mechanisms effectively protect against exploits.

To answer this question, the authors looked for vulnerabilities in 100 browser extensions, selected from the top 50 popular extensions on the Chrome Marketplace and 50 random ones. To find vulnerabilities, they performed blackbox testing as well as source code analysis. Every time they found a vulnerability, they exploited it to confirm that it was indeed a security problem. In total, 40 extensions had vulnerabilities. Interestingly, popular extensions are not more secure than the random ones.

Adrienne and her team then analyzed the vulnerabilities found and explained how the different security mechanisms did not help to prevent them. For example, in 7% of the cases, privilege separation did not help in blocking a content script exploit, and the extension was able to run code with full privileges. She then presented some techniques to mitigate the problems they found, such as allowing only HTTPS in cores, disallowing inline scripts, and forbidding evals. For each proposed mitigation, they evaluated how easy it would be to implement it, and how many extensions it would break. In the end, they proposed their mitigation techniques to the Google Chrome team, and they adopted three guidelines, which fixed 27% of the vulnerabilities they found.

Lior Malka (Intel) said that if a malicious extension could compromise the browser, all security mechanisms are gone, and having privileges separation in the browser wouldn't be a good idea. Adrienne replied that Google claims that it is hard to break the Chrome sandbox. David Brumley (CMU) asked why they found their vulnerabilities by using grep, instead of using more sophisticated static analysis techniques. Adrienne replied that there were no ready-to-use tools that understand the browser API architecture.

Establishing Browser Security Guarantees through Formal Shim Verification

Dongseok Jang, Zachary Tatlock, and Sorin Lerner, University of California, San Diego

Zachary Tatlock presented their work on designing a secure Web browser. Fully formal verification allows us to be sure that there are no bugs in the code; however, this approach has problems when applied to browsers. First, one must reimplement existing browsers from scratch, and second, it is not straightforward to specify what correctness is.

Instead, the authors propose a novel solution: isolate the browser's untrusted code in a sandbox, and create a "shim" that guards resource access. If the shim is formally verified, then we have guarantees that the security properties hold. Moreover, because the shim is small and does not evolve as a Web browser does, proving correctness of this component is more feasible.

Zachary explained their approach, which is composed of three parts. First, they adapt the untrusted browser code to run in a sandbox by substituting resource access with requests to the shim. Second, they design the shim itself. Third, they formally verify the shim to hold important security properties, such as response integrity, tab noninterference, and cookie integrity.

They implemented their approach in a browser called QUARK. For their untrusted code base, they used the WebKit framework. Zachary provided a demo of the browser to show that it works. In the future, they could add and verify new security properties.

Perry Metzger asked about the size of the code base and the proof. Zachary said that the shim is 750 lines of code, whereas the proof is 5000 lines. Charlie Rice (Google) asked about the challenges in specifying the security properties of a browser. Zachary replied that they spent a lot of time on it, adapting work on compiler implementation, then considered previous techniques that looked at model checking and adapted them to full formal verification. Kevin Borders (NSA) asked about the next properties they plan to verify formally. Zachary replied that same-origin policy is on their agenda.

The Brain

Summarized by Sarah Meiklejohn (smeiklej@cs.ucsd.edu)

Neuroscience Meets Cryptography: Designing Crypto Primitives Secure Against Rubber Hose Attacks

Hristo Bojinov, Stanford University; Daniel Sanchez and Paul Reber, Northwestern University; Dan Boneh, Stanford University; Patrick Lincoln, SRI

Hristo Bojinov explained that this work aims to build authentication systems resistant to "rubber hose" attacks, i.e., attacks in which security is broken by convincing (or coercing) users to reveal their passwords or other credentials. To avoid these types of attacks, users could therefore have passwords that they cannot in fact reveal consciously, but nevertheless know; this can be accomplished by focusing on procedural (how to ride a bike) memory, which is "implicit" in the sense that we can't retrieve things from it. To exploit this type of memory in forming a password, users are taught a given skill that they are then tested on in the process of authentication. The particular skill used by the authors is a game much like Guitar Hero: circles move down a screen that has a set of lettered regions at the bottom, and must be intercepted by typing the letter as the circle passes through the appropriate region. The speed at which the circles descend is calibrated so that the user should be able to succeed 70% of the time. Learning the password therefore consists of training on particular sequences, which takes 30–40 minutes, while authenticating involves playing the game with both the trained sequences and random (i.e., untrained) sequences and takes about 5–10 minutes; this is an example of a principle in neuroscience called serial interception sequence learning; how well the user does on the trained sequences relative to the untrained ones determines whether or not the user knows the password.

In terms of experimental setup, the authors first created sequences to avoid any bias or patterns; this meant picking sequences of length 30 with a uniform distribution over letters, as well as over pairs of letters (bigrams). Using a particular sampling mechanism (picking a random Euler cycle in a given graph), Bojinov argued that they obtained 37.8 bits of entropy, so for an adversary to predict a sequence was unlikely. They then used Mechanical Turk to obtain about 370 users. In addition to training the users on given sequences and then testing them later on those sequences, they also tested the users' explicit recollection of the sequence, i.e., showing them the sequence and asking them if they were familiar with it. Here, the trained sequences had a very small advantage over the untrained sequences (familiarity levels of 6.5 and 5, respectively, out of 10); when they tested trained against untrained sequences, however, they found that trained sequences had a 7–10% advantage immediately after training, while after one week they had an advantage of 6–7%. Finally, Bojinov discussed the difference between their work, in which the trained sequence really does function as a password, and related work such as keystroke timing, in which it wasn't clear how to change your habits; the related work therefore functioned more as a biometric than as a password.

Yossi Oren asked whether eye motions were correlated with these motor skills, and whether they had considered installing a camera at eye level to provide auxiliary information. Bojinov said he didn't know, but that they had considered other auxiliary data (e.g., EEG), and that there was lots of room to explore. Paul van Oorschot asked how they avoided revealing the entire sequence during authentication and thus potentially providing an attacker with the chance to train on it; Bojinov responded that users were explicitly not allowed to record the authentication session, which meant that it had to be supervised. Ian Goldberg asked how they chose the 70% hit rate. Bojinov responded that they had seen similar numbers in the neuropsychology literature, but that in general you didn't want it to be too high (as then a user who's just really good at the game will do about the same on untrained and trained sequences), and similarly for users who are not as good. Stefan Saroiu asked what he meant when he said passwords could be "reset." Bojinov replied that, unlike with a biometric, you could train a user with a different sequence and it would be like changing their password.

On the Feasibility of Side-Channel Attacks with Brain-Computer Interfaces

Ivan Martinovic, University of Oxford; Doug Davies, Mario Frank, and Daniele Perito, University of California, Berkeley; Tomas Ros, University of Geneva; Dawn Song, University of California, Berkeley

Ivan Martinovic explained how brain-computer interfaces (BCIs) such as EEG headsets might be used as an attack. BCI devices work by first obtaining EEG signals, which reflect the brain's electrical activity. In particular, EEG is

a measure of the brain's voltage fluctuation; while it provides high temporal resolution (in which we can see brain dynamics on the millisecond scale), its spatial resolution is low, and in particular depends on the number of points of contact. One additional problem with EEG data is that it is full of random artifacts: muscular activities, like blinking your eyes or closing your jaw, can cause interference, as can nearby power lines. To get rid of this noise, BCIs use signal processing techniques to attempt to match certain known patterns: e.g., alpha waves reflect a relaxed state, beta waves reflect active thinking, etc. In addition to examining raw EEG data, one can look at event-related potentials (ERPs); rather than a continuous stream of EEG data, these provide your brain's response to a certain stimulus. One such ERP that is well studied is called P300, which is the particular pattern associated with recognition and the classification of external stimuli as "personally meaningful." Now there are consumer-grade BCI devices that cost around $300; the particular one they looked at was the Emotiv EPOC device, for which third-party developers can design applications such as games.

The threat model the authors considered is therefore to trust the BCI device, but not to trust the third-party developers making the games. These developers nevertheless have access to raw EEG data, and thus can present certain stimuli and see how you react; in particular, using the P300 pattern, they can see whether certain stimuli are personally meaningful to you. To emulate the behavior of these developers, the authors obtained a developer SDK for the EPOC, and then considered how to discriminate between the targets (personally meaningful information) and the non-targets. They proceeded in two phases: a training phase, in which the targets are known and thus the patterns associated with these targets can be examined, and an online phase in which the targets are unknown. The training phase was, as the name suggests, used to train a classifier given the certain feature weights (i.e., the patterns associated with the targets); the target was then determined for the online phase using this classifier.

Their experimental setup consisted of 30 computer science students; each experiment lasted about 45 minutes per participant. There were two types of training: active, in which the user would (for example) count the number of occurrences of the number 6 in a flashing sequence of numbers, and passive, in which the user would be asked to recognize faces (again, in a sequence of flashing images). Each flashing sequence lasted 90 seconds and was meant to answer questions such as what is the first digit of your PIN?, where do you live?, what is your preferred bank?, what is your birth month?, etc. For all of these attacks they always did better than a random guess, and in most cases significantly so. For example, to determine your birth month, they could correctly guess the answer within the first six guesses nearly 100% of the time (as opposed to 50% with a random guess). So, although it might not be implemented (at least not for a wide audience) anytime soon, the authors concluded that BCI should be considered a potential side channel.

Tamara Denning asked why, given their threat model, they set up their experiments so that the images were not concealed. Martinovic responded that this was just a first step, and that doing so would require a more complicated setup. Peter Neumann asked whether they had thought about combating rubber hose attacks using this approach. Martinovic responded that there were related papers on using EEG data to detect coercion attacks. Luke Deshotels asked whether he expected these attacks to get easier as devices like the Kinect got better. Martinovic said that this was likely, although their work focused more on subconscious responses than facial ones. Paul Miller asked, if these devices were optimized for faces, whether they did in fact see emotional responses in their test subjects. Martinovic responded that in fact the classifier limited frontal influence, so they didn't see facial expressions. Finally, Hristo Bojinov asked whether they had looked at lie detector designs. Martinovic responded that P300 had been considered for a new generation of lie detectors, which they cite in the related work, but that their threat model in this paper was slightly different (e.g., in an interrogation the person knows they are being interrogated, unlike here).

Rump Session

Summarized by AbdelRahman M. Abdou (abdou@sce.carleton.ca) and Saran Neti (saran@ccsl.carleton.ca)

Bryan Parno (MSR) presented a new technique called "Quadratic Span Programs" (QSP), which helps verify arbitrary computation functions. The technique provides a new characterization of NP. It has a linear structure and supports efficient cryptographic applications. The technique is implemented using elliptic curve cryptography.

Jeremy Epstein (NSF) talked about the Secure and Trustworthy Cyberspace (SaTC) funding program by the NSF. The program offers four different types of grants: small (500k/3 years), medium (1.2M/4 years), frontier (10M/5 years) and an education grant (300k/2 years). SaTC is interdisciplinary, ranging from behavior science to mathematical science. Proposals are peer reviewed and responses take 90 days. Proposals must clearly indicate the threat adversarial model. When Epstein's time slot approached completion, the session chair made a joke saying he could take extra time as he was from the NSF.

The SaTC program can be found here: http://www.nsf.gov/publications/pub_summ.jsp?ods_key=nsf12596.

Cynthia Sturton (UC Berkeley) presented "Verification with Small and Short Worlds," pointing out the importance of correct and secure virtualization. She discussed challenges with exploring the large state space, including isolation properties in hypervisors and emulators, the need to verify page tables, caches, memories requiring large data structures, etc. They published this work in the Formal Methods in Computer-Aided Design (FMCAD) 2012 conference.

Ian Goldberg (University of Waterloo) presented "A Unified Notation for Elliptic Curve Protocols," which aims to decrease ambiguity due to notations used in elliptic curve protocols. He added a sense of humor to his presentation by asking two volunteers to perform short ballet movements mapping to his proposed notations.

Zachary Peterson (Naval Postgraduate School) presented d0x3d, a tabletop network security game designed to teach students network security terminologies and basic computer security fundamentals. Four players take the role of an elite hacking syndicate. The objective is to infiltrate a network and escape with four digital assets. The game can be found at: http://www.d0x3d.com/.

Sadia Afroz (Drexel) presented "Authorship Attribution of the New York Times Hoax." Motivated by a New York Times columnist whose name was used in a hoax about Wikileaks, the authors worked toward attributing authorship in the presence of an adversary. They published this work in the IEEE S&P Symposium 2012.

Jeremy Clark (Carleton University) presented a technique to establish a secure end-to-end SSL/TLS connection through a proxy server. He proposed a modification to the HTTPS handshake scheme to include the proxy configuration. His objective is a secure end-to-end connection that includes a proxy server in the middle.

Roger Dingledine (Tor Project) presented some statistics about the current status of the Tor Project in terms of bandwidth, the number of exit nodes, relay nodes, etc. He also discussed how hard it is to get people's cooperation and increase the number of fast exit nodes, which is crucial for measuring diversity. He added that five exit nodes are chosen 20% of time and about 50 are chosen 50% of time. The Tor Project recently got a new NSF grant.

Ziye Yang (EMC Labs) presented "Exploring VM-based I/O Performance Attacks in the Public Cloud," an architecture for a distributed I/O measurement framework. Their goal is to provide a third-party tool for measuring the VM disk I/O SLA from a tenant view and explore VM disk I/O-based performance attacks.

Eric Wustrow (University of Michigan) shared some of the funny insights that they came across while scanning the Internet for factorable public keys. For example, with a classic Google search, he showed that some people post their private keys online. He used as an example a person who embedded some random text in his private key to baffle whoever is searching for it; however, the strings "begin RSA private key" and "end RSA private key" were repeated in the file, which raised his rank in Google's returned results.

Erika Chin (UC Berkeley) measured user confidence in smartphone security and privacy, and presented the findings: people don't trust their phones. A generic system design challenge would be to alleviate users' concerns.

Tamara Denning (University of Washington) presented a board game, Control-Alt-Hack, which is designed to introduce players to white-hat hacking, elaborate some computer security aspects, and, above all, be interesting to play. The game can be found at: http://www.controlalthack.com/

Gilbert Milhouse (RTI International) announced senior cybersecurity job vacancies at his company.

Alex Halderman (University of Michigan) showed how state elections and voting system in states that deliver ballots by mail (such as Washington) can be hacked. He demonstrated this in less than five minutes live by picking a Washington-resident from the audience and searching for his data on Washington State public information Web sites. He managed to get his birthday, driver's license number, and finally arrived at a page where he could change the volunteer's mailing address! Conclusion: don't use public/easy-to-obtain information to change the mailing address.

Adrian Mettler (UC Berkeley) presented "Reviewable Retrofit of Legacy Web Applications for XSS Prevention," addressing security issues with Web templates such as untrusted content being added to the Web output and auto-escaping template systems escape by default, etc. He then proposed their approach, "Security-Effort Tradeoff," and showed how they evaluated it.

Hannah Pruse (University of Oregon) presented "Host Identification via USB Fingerprinting," a methodology that aims to fingerprint a machine using USB communication data. In the presentation, Pruse pointed out the importance of determining a computer's identity for performing secure transactions. She presented methodologies and their evaluation results.

David Barrera (Carleton University), along with Ildar Muslukhov and Yazan Boshmaf (University of British Columbia), showed how to circumvent the face-unlocking scheme employed by Google in Android Jelly Bean. They demonstrated this live on a Galaxy Nexus smartphone running Android 4.1 Jelly Bean. Through a simple Facebook search for the victim's photo (Yazan) and some photo-editing tricks on Paint.NET, they succeeded in displaying Yazan blinking on the screen which should unlock his phone. While the unlocking demonstration failed, the "animated" blinking photo, created in just a few minutes, was hilarious.

Daniel J. Bernstein (University of Illinois at Chicago) discussed how cryptography doesn't always provide what it promises, and that we should stop blaming the user constantly. Their project is NaCl: Networking and Cryptography library, a cryptographic library that eliminates failures. It can be found at http://nacl.cr.yp.to/.

A Chance of Clouds

Summarized by Aaron Blankstein (ablankst@cs.princeton.edu)

Whispers in the Hyper-Space: High-Speed Covert Channel Attacks in the Cloud

Zhenyu Wu, Zhang Xu, and Haining Wang, The College of William and Mary

Zhenyu Wu began by explaining that a major information security and privacy concern for VM co-residency is high bandwidth side-channel attacks; however, on virtualized x86 systems, covert channel attacks have not yet proven to be practical, and thus the threat is widely considered a "potential risk." Previous work demonstrated only low bandwidth side channels. These researchers presented a novel covert channel attack that is capable of high bandwidth and reliable data transmission in the cloud. The presenter first explained that classic covert cache channel schemes perform very poorly on virtualized platforms because of indirection in addressing, uncertainty in scheduling, and sometimes physical limitations (VMs may not share the same cache).

He then demonstrated how the addressing and scheduling obstacles can be overcome by encoding data purely through timing patterns on the L2 cache. Then, to overcome the physical limitations of a shared cache, the researchers exploited atomic instructions to use the memory bus as a covert channel medium. Further, they implemented a robust communication protocol, and demonstrated realistic covert channel attacks on various virtualized x86 systems. Experiments showed that covert channels do pose serious threats to information security in the cloud. Finally, the presenter discussed some possible ways to mitigate covert channels in virtualized environments.

Bill Brumley (Qualcomm) asked if the VM location problem (VMs that do not share caches) could be solved by running one sender and multiple receivers spread out among cores? Zhenyu Wu responded that in order to do that, the attacker would have to generate many instances residing on the same hardware. In their tests, just to spawn a pair of co-residing VMs, the researchers had to spawn a couple hundred VMs. Had they given any consideration to using the instruction cache as a side channel? They had not, but that might be a good channel to attack.

Policy-Sealed Data: A New Abstraction for Building Trusted Cloud Services

Nuno Santos, MPI-SWS; Rodrigo Rodrigues, CITI/Universidade Nova de Lisboa; Krishna P. Gummadi, MPI-SWS; Stefan Saroiu, Microsoft Research

Nuno Santos explained that mismanagement of cloud software by administrators poses a serious threat to the integrity and confidentiality of customer data hosted by cloud services. Trusted computing provides an important foundation for designing cloud services that are more resilient to these threats. However, current TPM chips are ill-suited to the cloud as they expose too many internal details of the cloud infrastructure, limit VM and data migration across cloud nodes, and perform poorly.

The researchers presented Excalibur, a system that addresses these limitations by providing a new trusted computing abstraction called policy-sealed data. This allows data to be sealed (encrypted to a customer-defined policy) and then unsealed (decrypted) only by nodes whose configurations match the policy. To provide this abstraction, Excalibur uses attribute-based encryption, which reduces the overhead of key management and improves the performance of the distributed protocols employed. To demonstrate that Excalibur is practical, researchers incorporated it in the Eucalyptus open-source cloud platform.

Paul van Oorschot (Carleton University) commented that this work was similar to IBM control vectors. Santos responded that this work used the same insight but used it to overcome the limitations present in the cloud. Trent Jaeger (Penn State) asked whether the data could be sealed with policies providing stronger integrity guarantees such as the Biba or Clark-Wilson models. Santos responded that the system provides a way to bootstrap trust in the cloud. Then, once a secure hypervisor is securely loaded, stronger properties can be achieved.

STEALTHMEM: System-Level Protection Against Cache-Based Side-Channel Attacks in the Cloud

Taesoo Kim, MIT CSAIL; Marcus Peinado and Gloria Mainar-Ruiz, Microsoft Research

Taesoo Kim explained that cloud providers share physical resources to support multi-tenancy of cloud platforms. However, the possibility of sharing the same hardware with

potential attackers makes users reluctant to offload sensitive data into the cloud. In fact, side-channel attacks via shared memory caches have been demonstrated to break full encryption keys of AES, DES, and RSA. Kim first discussed a strawman solution to mitigate the cache side channel. In this solution, each VM receives private pages, such that only a particular VM can use memory locations that map to these private pages. Unfortunately, this solution fails to efficiently use available memory.

Kim presented their solution, STEALTHMEM, which manages a set of locked cache lines per core. These locked cache lines are never evicted from the cache, and STEALTHMEM efficiently multiplexes them such that each VM can load its own sensitive data into the locked cache lines. Thus, any VM can hide memory access patterns on confidential data from other VMs. Unlike existing state-of-the-art mitigation methods, STEALTHMEM works with existing commodity hardware and does not require profound changes to application software. The researchers showed that STEALTHMEM imposes 5.9% of performance overhead on the SPEC 2006 CPU benchmark, and between 2% and 5% overhead on secured AES, DES, and Blowfish, requiring only between 3 and 34 lines of code changes from the original implementations.

Stefan Saroiu (Microsoft Research) commented that because x86 has no support for locking cache lines in hardware, this had to be implemented in software, meaning that every memory operation needed to be interposed upon. Kim responded that this was not the case, because the private page is not accessible from other VMs.

Embedded Security

Summarized by Shane Clark (ssclark@cs.umass.edu)

Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices

Nadia Heninger, UC San Diego; Zakir Durumeric, University of Michigan; Eric Wustrow and J. Alex Halderman, University of Michigan

▶ ***Awarded Best Paper!***

Zakir Durumeric presented this work investigating the state of public-key cryptography on the Internet. He began by pointing out the long history of security vulnerabilities stemming from poor random number generation. Durumeric noted that many of these vulnerabilities were only found after close examination of specific implementations. In contrast, Durumeric and his co-authors chose to examine vulnerabilities based on the entire set of publicly available keys on the Internet that use TLS and SSH. They gathered this set by scanning the entirety of the IPv4 public address space for hosts with ports 22 (SSH) or 443 (HTTPS) open. They then performed handshakes with all available hosts in order to retrieve each of their public keys.

Durumeric et al. looked for both repeated keys and repeated factors in keys. They found that 5.6% of TLS hosts and 9.6% of SSH hosts were inappropriately sharing keys. Additionally, they found that repeated factors in RSA keys and repeated "ephemeral keys" used in the DSA algorithm allowed them to factor 0.5% of all TLS keys and 1.03% of all SSH keys. The majority of these repeated and weak keys were generated by embedded devices that either use default keys or generate keys on first boot. The underlying cause of the weak or repeated keys generated on first boot is the widespread use of /dev/urandom as a source of entropy for Linux-based operating systems, despite the documentation specifically warning against this application. /dev/urandom is largely or completely deterministic for embedded devices at first boot because services query it for randomness before there is time for sufficient randomness to accumulate. The recommended source of randomness, /dev/random, goes largely ignored because it requires a blocking call with undefined return time.

After completing their analysis, the authors contacted about 60 companies about the issues they uncovered. Approximately 20 companies responded, with the rest ignoring their communications. Only three of the companies have informed the authors of security advisories in response. Seeking to mitigate the source of the problem, the authors also worked with the Linux kernel team to add new sources of randomness, including interrupts and unique (though deterministic) hardware identifiers such as MAC addresses. Finally, the authors created an online service that allows users to check the quality of their keys.

Perry Metzger asked for Durumeric's opinion on seeding embedded devices with deterministic, unique seeds at manufacture time. Durumeric answered that it might help but that there are pros and cons to that approach. Ian Goldberg asked if the authors found any non-prime numbers used to generate keys. Durumeric responded that they did not find any, but they did find a number of very small prime factors.

TARDIS: Time and Remanence Decay in SRAM to Implement Secure Protocols on Embedded Devices without Clocks

Amir Rahmati and Mastooreh Salajegheh, University of Massachusetts, Amherst; Dan Holcomb, University of California, Berkeley; Jacob Sorber, Dartmouth College; Wayne P. Burleson and Kevin Fu, University of Massachusetts, Amherst

Amir Rahmati presented this work that leverages SRAM decay to mitigate key recovery attacks against batteryless embedded devices with no real-time clock (RTC). Rahamti noted that many people carry such devices in their wallets, including transit cards, passports, contactless credit cards, and employee IDs. There are several published attacks against these devices that rely on the attacker's ability to

query the device many times rapidly and then extract keys via timing/power analysis or brute force. TARDIS addresses these classes of attacks, which Rahmati referred to as "semi-invasive," but cannot offer protection against adversaries with physical access to devices.

TARDIS uses the predictable rate of SRAM decay to estimate how much time has elapsed between power-up cycles for embedded devices without the need for additional hardware. SRAM is volatile memory that will rest in an unknown state of randomness after losing power. When all cells in an SRAM bank have reached the resting state, there will be an approximately equal number of 0s and 1s. TARDIS estimates time by initializing a section of memory to the same value and measuring the distribution of 0s and 1s after a power outage. Because of the inconsistent rate of decay, TARDIS cannot return an exact estimate in all cases. Instead, it can identify very short periods of power loss, return an estimate for moderate periods of power loss, and identify when the elapsed time is greater than a maximum threshold. In practice, TARDIS could throttle rapid queries by refusing to respond if a minimum threshold time has not yet elapsed since the last query. The time constants for TARDIS are determined by the capacitance in the circuit and are also affected by changes in temperature, which could be compensated for on devices that integrate a temperature sensor.

Ian Goldberg asked about the implications of attacks that bombard the device under attack with ion beams. Rahmati answered that TARDIS does not address adversaries with such sophisticated tools. Perry Metzger asked about the feasibility of an attack where the adversary rapidly heats the device and then cools it before waking it back up. This would lead TARDIS to overestimate elapsed time because the temperature sensor would not observe the high temperature. Because chip surface is small, Metzger thought that this might be feasible. Rahmati responded that they have not done thermal transfer calculations to estimate the efficacy of this attack, but he believes it is likely to be difficult to mount in practice.

Gone in 360 Seconds: Hijacking with Hitag2

Roel Verdult and Flavio D. Garcia, Radboud University Nijmegen; Josep Balasch, KU Leuven ESAT/COSIC and IBBT

Roel Verdult presented this paper exposing failings in the Hitag2 vehicle immobilizer. Vehicle immobilizers are passive RFID tags integrated into the keys of modern automobiles to prevent hotwiring. They are legally required for new cars sold in Europe, Australia, and Canada. The Hitag2 is the most common immobilizer and is used by at least 34 car makes. Many newer cars also use Hitag2-based systems to implement keyless ignition. According to the manufacturer's (NXP) Web site, Hitag2 uses a mutual authentication protocol and offers "unbreakable security." The Hitag2 protocol was reverse engineered and posted on the Internet in 2007. Verdult and his co-authors used this information to identify three weaknesses in the protocol. First, the tag's initial challenge does not include a nonce, making the protocol vulnerable to replay attacks. Second, the cipher uses only a 48-bit secret key and the reader's nonce is only 32 bits, reducing the strength of the cipher further. Finally, weaknesses in the filter function used internally by the cipher result in partially deterministic output for 1/4 of all communications. Using these weaknesses and a protocol weakness that allows an attacker to use the tag as a keystream oracle, Verdult et al. were able to mount three successful attacks against Hitag2 systems. The most powerful of these attacks allowed them to bypass the Hitag2 systems on more than 20 cars in practice, each time taking fewer than six minutes. Verdult concluded by noting that NXP confirmed the researchers' findings and worked constructively on mitigations. Unfortunately, the cipher is irreparably broken.

Jeremy Epstein asked at what point the cost of the technology required to steal a car makes it a more attractive option than attacking the owner with a wrench. Verdult answered that the device they used cost less than $200 and that attacking a car's owner directly attracts more attention and reveals exactly how the car was stolen. Epstein followed up by asking if Verdult thought that NXP's reaction would be different if the device cost $0.50 and was easily available. Verdult responded that the attacks only improve over time, as evidenced by the Mifare Classic smartcard, but that the manufacturer will always claim that the attack is difficult and impractical while they are able to do so.

Secure Computation and PIR

Summarized by Benjamin Mood (bmood@cs.uoregon.edu)

Taking Proof-Based Verified Computation a Few Steps Closer to Practicality

Srinath Setty, Victor Vu, Nikhil Panpalia, Benjamin Braun, Andrew J. Blumberg, and Michael Walfish, The University of Texas at Austin

Srinath Setty talked about how to verify a remote computation which is performed by another party like a cloud. The ideal situation would be for the cloud to return some sort of data along with the computation result which could be used to verify the computation. This data is considered to be a proof of correctness. The system which was presented is called GINGER. In this system, the server takes a computation in the form of a circuit and executes it. A proof of correctness is also returned along with the result of the computation. Each entry in proof of correctness corresponds to a circuit. The client then runs a set of tests on the proof to verify the results. If the tests pass then the client accepts the result. In GINGER the client uses some randomness to determine which parts of the proof to verify so it is possible

for a wrong entry in the proof to not be detected. GINGER contributes the following: it eliminates some of the tests the client has to perform on the returned proofs, it changes the model of verified computation from arithmetic circuits to systems of equations, and it also adds support for floating point numbers. These improvements decrease the end-to-end cost for verifying computations.

Srinath explained how solving a system of equations in their system is like executing a program. They created many smaller constructs, such as X != Y, in the form of systems of equations. They then built a compiler which transforms a program to a system of equations using the smaller pieces they created. GINGER's representation is much smaller than that of a Boolean circuit. Srinath noted that one limitation of their compiler is that the number of iterations in a loop must be known at compile time. They used matrix multiplication as their benchmark. In this system, the client cannot gain from outsourcing a single instance of the computation. There is a minimum number of instances which outsourcing the computation will be beneficial.

Srinath said GINGER reduces the computation time from past systems by a factor of 10^23, but they still need a reduction of 10^3. He noted this reduction may be possible to do in the very near future. GINGER supports a GPU-based implementation, a high-level language, and a compiler that compiles from the high-level language to their execution model. Srinath said they reduced the amount of work the client has to do, but the cost for the server is still high.

Someone asked whether this system was vulnerable if the server wants to forge an answer and a proof. Srinath answered that in their system the server could be malicious, but there are no guarantees about the security of the network. Dan Boneh noted that their solution appears to be dependent on multiple rounds and asked if a single round protocol is possible. Srinath answered that it is possible but right now they use an interactive proof, and he was not sure if there is a non-interactive version.

Optimally Robust Private Information Retrieval

Casey Devet and Ian Goldberg, University of Waterloo; Nadia Heninger, University of California, San Diego

Casey Devet presented on private information retrieval (PIR). He explained how it solves the following situation: suppose a person wants to get data from a database but does not want to leak any information about which record he is interested in retrieving. The naïve way to do this is to download the whole database. Casey presented a way for retrieving a block of data from a server without the server learning any knowledge about which block is retrieved. One of the past solutions for this problem is Goldberg's scheme, which uses Shamir secret sharing. In this method, L servers are used and the data is shared across the servers. All the client has to do is polynomial interpolation to find the result once the client gets the information from the servers. The client does not need responses from all the servers, only enough to do polynomial interpolation. It is possible to retrieve the data with some incorrect responses from servers. The most incorrect responses the client can receive and still perform polynomial interpolation is V. There is an assumption in this method that no more than T servers are colluding. Casey listed K as the number of servers responding.

The algorithm Casey presented, Cohn-Heninger, uses fast lattice reductions and runs with a V of at most (K – √K * T)) with single polynomials. He then explained how they have a portfolio algorithm which chooses the best state-of-the-art algorithm depending upon K, T, and V. The portfolio algorithm picks the fastest algorithm for the requirements of the current request. Casey explained that if the client only asks for a single block of information, the current algorithms are optimal. However, if multiple blocks are requested, then it is possible to decode multiple blocks at the same time through a variation of their algorithm, which also allows for an improvement in robustness. In this algorithm, they need at least T + 1 servers to be honest. He described how it is a multi-polynomial variant that runs with a V of at most (K – T – 2). This algorithm requires clients to randomize queries. Casey noted this algorithm fails a small probability of times. Fails, in this case, means the algorithm cannot recover a block. The system was implemented in Percy++.

Casey concluded that with their system, PIR can be done efficiently even in the presence of adversaries of either the client or server. He noted it should even be feasible to use their system on mobile devices. Casey provided two main takeaways for this work. The first was the improved robustness of the algorithm to the optimal bound and the second was that their solution is fast.

Nikita Borisov said he was confused about the graph presented, which showed the different algorithms and the time they took. He asked why the single optimal polynomial algorithm is faster when it is based on brute force. Casey answered that when dynamic programming is used it is a couple of orders of magnitude faster. Nikita Borisov then asked if V was the number of servers which are actually misbehaving? Casey answered yes.

Billion-Gate Secure Computation with Malicious Adversaries

Benjamin Kreuter, Abhi Shelat, and Chih-hao Shen, University of Virginia

Chih-hao Shen presented his team's state-of-the-art two-party secure computation system which allows for very large

circuits. In two-party secure computation, two parties can send input and receive input from a function without either party gaining any information about the other's inputs or outputs. This function acts as a trusted third party. The standard two-party secure computation model accounts for semi-honest adversaries and is based on the Yao protocol. There are two main components to the Yao protocol. The first is an oblivious transfer (OT). The OT allows a user to select one and only one of two numbers which the second party has without the second party finding out which number was selected. The second component is the garbled gates. Garbled gates are similar to a typical Boolean gate but have differences in the way the truth table is created. In a garbled gate, each input wire has two random values representing 0 and 1. The input wires are the keys to decrypt the correct output wire value. It was noted this system also runs in the malicious model.

Chih-hao noted there were two main challenges to their work. The first challenge was how to create and handle large circuits. The second was how to speed up the evaluation protocol. They created a compiler which could create larger circuits than the past compilers. Past compilers needed massive amounts of memory to create the circuits. Chih-hao described how their compiler was able to create circuits of billions of gates. The largest circuit the presenters were able to compile using Fairplay, one of the past compilers, was 50,000 gates. He also noted the speed of the compiler was faster; their compiler was able to create an AES circuit in one second as opposed to 12 minutes with the Fairplay compiler.

Chih-hao explained how they able to calculate 432,000 gates per second in their system during his tests. 154,000 of those were non-XOR gates. The amount of non-XOR gates matters since XOR gates are evaluated "for free." Their system incorporates various optimizations, including cut and choose, input consistency, selective failure, output authentication, free XOR, garbled-row reduction, and random seed checking. Chih-hao emphasized that the most important technique they incorporated into their system was parallelization of the cut-and-choose protocol for the malicious model. In this protocol, with sufficient bandwidth and sufficient cores, the time difference between semi-honest (1 + C time) and malicious (1 + C + e time) threat models is insignificant. They tested three versions of the system. The baseline system was based on HEKM's pipelined execution system. The second version was their system with a priority on end-to-end time. And the third version of their system had a priority on communication bandwidth.

Nikita Borisov wanted to clarify if garbling and compilation is the same thing? Chih-hao responded that garbling has to be done for each execution. Lior Malka noted that in any implementation of secure two-party computation, the user writes in a high-level language and the compiler compiles down to a circuit. So if a user wants to do a search in a database, then the user would have to recompile—since the size of the database might change—so recompiling is not a one-time cost. So why is it important to have a compiler? Chih-hao responded that if there is a compiler, then users can use a high-level language which is easier to use.

Invited Talk

Summarized by John-Paul Verkamp (verkampj@indiana.edu)

Cyber Security Education in a Time of Change

Cynthia E. Irvine, Naval Postgraduate School, Monterey, CA

Cynthia E. Irvine spoke on cybersecurity education and, more particularly, about what she feels needs to be done to keep up in an ever-changing world. She particularly made the case for a difference between education and training, citing that in the case of training, you are focusing on repetitive behaviors that do not require much in the way of creativity. While this is adequate for operators and maintainers, a deeper understanding is necessary for those implementing new products and services. In particular, she believes that security should be a part of even high school computer science classes up through all levels of post-secondary education.

To that end, she spoke about a program from the National Security Agency called the National Centers of Academic Excellence in Cyber Operations program. This program is designed to make security a first-class objective in computer science, computer engineering, and electrical engineering programs around the country. The program requirements are rather strict, including everything from low level programming (C or even assembly), hardware modeling, and reverse engineering up through principles of networking, operating systems, and databases, all with a firm grounding in discrete mathematics, statistics, and calculus, with at least some touch of security included. In one question after her talk, she did acknowledge the difficulty in adding security to a course curriculum where the teachers themselves may not be familiar with security issues, but she reiterated how important it is to cover at least basic topics in each course so that courses dedicated to security can move on to more complicated matters.

One especially interesting example she gave of the kind of coursework that could better prepare students for cybersecurity was a practical experiment where students were told that they would have to write down the first 100 digits of pi while at the same time being told that they would be expected to cheat—but getting caught would still lower their grade. This gave students a taste of the creativity that attackers in cyberwarfare could bring to the table in a way that traditional defensive examples rarely do.

She concluded her talk with a short summary of fellowship programs run by the US government which offer funding in exchange for the same number of years of government work after graduation. In particular, she mentioned the SMART and ISAP programs, both run by the Department of Defense.

Authentication and Secure Deletion

Summarized by Blase Ur (bur@cmu.edu)

Progressive Authentication: Deciding When to Authenticate on Mobile Phones

Oriana Riva, Microsoft Research; Chuan Qin, University of South Carolina; Karin Strauss and Dimitrios Lymberopoulos, Microsoft Research

On modern mobile phones, a typical user walking down the street would be asked to authenticate many times, leading some users to entirely disable authentication. The authors posit that a progressive authentication scheme using multiple sensor readings to continually authenticate a user could maximize the tradeoff between security and usability. In this talk, Oriana Riva presented a scheme that decides when to ask a smartphone user to authenticate, and for which applications.

The phone will compute a "user authenticity level" over time using sensors including face recognition, proximity to known devices, and the continuity of touch. Applications fall into three security levels: confidential (e.g., banking), private (e.g., text messaging), and public (e.g., weather forecasts). If a user tries to open an application whose security threshold is above the currently computed authenticity level, the phone will ask for a PIN. The signals used for sensing are fed to an SVM machine-learning model. In addition to the general SVM model, a personalized model is trained on the user's own phone to recognize the particular user.

Rivera discussed a nine-participant user study employed to evaluate the scheme, finding the system reduced authentication overhead by 42% without allowing unauthorized accesses during the study. She further discussed implementation enhancements to reduce power consumption, including offloading some processing of sensor readings to the cloud.

Questions and discussion initially focused on how the SVM classifiers for the machine-learning model would be trained. In particular, Rivera discussed how classifiers could be retrained through software updates since only a decision tree would need to be downloaded. She further clarified that parts of the model are user-agnostic, whereas face and voice recognition elements do need to be trained for a particular user. Logan Gabriel (IBM) questioned whether privacy-concerned users might not want their voice and face patterns uploaded to the cloud for analysis. Rivera responded that the scheme does not require processing to happen on the cloud; instead, processing could happen on the phone or on a user's own PC.

Origin-Bound Certificates: A Fresh Approach to Strong Client Authentication for the Web

Michael Dietz, Rice University; Alexei Czeskis, University of Washington; Dirk Balfanz, Google Inc.; Dan S. Wallach, Rice University

On the Internet, authentication schemes generally provide either a familiar user experience or the protection of credentials against active attackers, but not both. In current practice, a client would send a username and password to a server over TLS and receive back a cookie, yet a spate of recent attacks against certificate authorities have brought into question whether this approach actually protects against man-in-the-middle (MITM) attacks. To thwart potential MITM attacks, a user could also present a certificate known to the server in place of a password, yet this is an unfamiliar user experience.

Michael Dietz presented the idea of TLS Origin-Bound Certificates (OBCs), which are unique in that there is one client certificate per origin created on the fly by a user's browser. These self-signed certificates identify a TLS channel rather than a user, preventing MITM attacks without introducing an unfamiliar user experience. In the end, the server binds cookies to information contained in the OBC, effectively attaching the cookie to the particular channel. Following the initial authentication process, information from an OBC must always be presented in tandem with the relevant cookie.

Brad Hill (PayPal) questioned the authors' threat model. He proposed that an attacker in possession of a certificate that appears genuine to a user could use forgery-signed false code to cause the user to make requests to the server. Dietz confirmed that this type of attack would be successful. Stuart Schecter (Microsoft Research) noted that the authors' scheme prevents a cookie from being sent to a server with compromised certificates and wondered whether one could instead calculate the hash of the certificate that the cookie can be sent to, which Dietz agreed would provide some of the same guarantees as OBCs. Tom Ristenpart (University of Wisconsin) pointed out that OBCs require modifications to the TLS handshake. Dietz clarified that the OpenSSL implementation on both the server and client, as well as the network stack on the client, had been modified.

Data Node Encrypted File System: Efficient Secure Deletion for Flash Memory

Joel Reardon, Srdjan Capkun, and David Basin, ETH Zurich

For storage devices that store data in blocks, secure deletion can usually be accomplished by overwriting sensitive data. However, log-structured file systems, which are ubiquitous in flash memory since erasures are quite costly on a physical level, present significant challenges to secure deletion. In these systems, changes are simply appended to a log without erasing the original data. In this talk, Joel Reardon presented

the Data Node Encrypted File System (DNEFS), designed to maximize the ratio of bytes deleted in secure deletion to full blocks of flash memory erased.

The idea behind DNEFS is to encrypt every data node (atomic read/write unit) with a unique key. Keys would be collocated in a densely packed key storage area. Periodically, unused keys could be purged from the key-storage area, rendering the encrypted node securely deleted. There would be a tradeoff knob for the frequency of purges versus the lifetime of data intended to be deleted. Reardon also described how DNEFS had been implemented as a single patch on top of Linux's UBIFS. This implementation was observed to run normally on a Nexus One phone, yet was a bit slower than standard UBIFS since it was always necessary to read the keys from the storage medium.

Mike Freedman (Princeton) noted that DNEFS only periodically provides guarantees about secure deletion and further questioned the assumptions of the work, such as whether the size of the erased data is usually smaller than the erase block. Reardon responded that the DNEFS approach is always smaller than the naïve solution since the key-storage area is only about 0.5% of the file system, and thus less data needs to be erased. Albert Wong (Google) wondered whether the authors had considered creating a virtual block device on top of DNEFS, which Reardon noted was considered, yet would have lacked features, including the ability to rapidly mount a drive following a crash. Dan Farmer questioned both whether the keys are kept in memory and what would happen if the key-storage area were corrupted. Reardon explained that the keys are kept in one particular buffer of RAM that is poisoned after use, and he agreed that a corruption in the key-storage area would be problematic unless the keys were duplicated. Logan Gabriel (IBM) referenced another Security '12 paper in questioning the randomness in the creation of keys in embedded devices, and Reardon responded that an accelerator could be used to reliably generate randomness.

Invited Talk

Summarized by John-Paul Verkamp (verkampj@indiana.edu)

Life as a Target—Welcome to the Club

Dickie George, Johns Hopkins Applied Physics Laboratory

Dickie George's talk about Life as a Target underscored just how much the world has changed in the decades since he himself first became a target. Originally, you could trust foreign spies, at least to the point that you knew what they were looking for and what they would do to get it. When you were a target, it was because you signed up for it and you knew (or at least had been told) what you were getting into. In today's world, the rules have become much more nebulous and people do not need to sign up to be targets. The Internet has become so pervasive that just signing up for online services is enough to provoke attacks.

Throughout his talk, George recalled a number of stories about others who also lived as targets. He could tell if Victor, ex-KGB, was in a particularly suspicious mood by how he sat at a restaurant. If he sat with his back to the wall, he was feeling relatively carefree. If he were feeling more paranoid, he would purposely sit with his back to the room and watch everything in the reflection of George's glasses so that he could watch without being observed in turn.

In another instance, George recalled being asked to prepare a speech for the director of the NSA to give at Black Hat. Upon delivery, however, George was informed that the director would not be giving the presentation after all—he would. Why? Because information from a credible source had implied that there was a chance of an assassination attempt if the director showed up. So they had weighed the options and chose to send someone who was less of a target.

After his talk, Dickie George answered quite a few questions. When asked if he believes it is actually possible to make computers secure (without just sawing the processor in half), he responded that it is the people who tend to make systems less secure. He also said that in the old days, it used to be possible to design systems smart enough to compensate for their users. Now computers are just too complicated. Therefore, in today's world, we need better education. People need to have at least some understanding about the security underlying the systems that they are using.

Someone asked whether the big issues of security are even possible from perspectives of those in academia or business or if only governments truly have the resources to make this work. His response was that may have been true in decades past where the government had a 20-year head start in technology, but in today's world consumer technology has caught up. In reality, all three—government, academia, and business—are necessary to deal with the modern world.

One final questioner asked what could possibly be done from a corporate perspective when your coworker is in China, your boss is in England, and his boss is in South America. In this case, Dickie George's response was that we need to work together to ensure that the focus is on defense versus offense rather than on this country versus that one. He also stated that he is perfectly all right with making the Chinese network more secure if that means ours is as well.

Privacy Enhancing Technologies and Network Traffic Analysis

Summarized by Blase Ur (blaseur@gmail.com)

Throttling Tor Bandwidth Parasites

Rob Jansen and Paul Syverson, US Naval Research Laboratory; Nicholas Hopper, University of Minnesota

On the Tor network for anonymous communication, a small number of connections using BitTorrent can create the majority of traffic on the network. In this work, Rob Jansen proposed approaches for throttling these bandwidth parasites to improve performance for the majority of users of the network. Tor clients currently create a single TCP connection to a guard node for their first hop in the network, and the authors propose throttling the client at this point. Guards must figure out which clients to throttle and at what rate using only local information and without requiring tweaking over time.

In his presentation, Jansen proposed three algorithms for accomplishing this goal: bit-splitting, flagging, and threshold. The authors' bit-splitting algorithm would split a connection between n clients by throttling each client at a maximum of 1/n of the maximum rate. Their flagging algorithm attempts to identify bulk connections and throttle those aggressively. Finally, their threshold algorithm adjusts both the throttle rate and the selection of connections to throttle in order to throttle the loudest connections. They evaluated these schemes using Shadow, a Tor simulator, finding that all three algorithms were successful at throttling bulk users.

Algis Rudys (Google) suggested that there are some seemingly legitimate, yet high-bandwidth, uses of Tor, such as Skype communication. Jansen agreed that not being able to differentiate between different types of high-bandwidth behavior is a weakness of the scheme, yet necessarily follows from the design of Tor. Ian Goldberg (University of Waterloo) suggested that an evil client could be its own entry node to evade this technique, which Jansen agreed was true. Roger Dingledine (Tor) expressed happiness that the team evaluated their protocols on the Shadow simulation system and noted that he's looking forward to actually implementing these approaches on the real Tor network. He also noted that, due to Tor's infrastructure, it is challenging to determine how to measure whether or not these approaches are working in the wild.

Chimera: A Declarative Language for Streaming Network Traffic Analysis

Kevin Borders, National Security Agency; Jonathan Springer, Reservoir Labs; Matthew Burnside, National Security Agency

Behavioral analytics are often a necessary technique for analyzing network traffic, enabling the detection of attacks like side-jacking. However, the authors believe that even security professionals have difficulty translating from the declarative statements in which they think to the procedural code required by Bro, a common IDS. Kevin Borders presented Chimera, a declarative language that would be converted to Bro code for analyzing streaming traffic. The structure of Chimera is loosely based on SQL, yet adds additional features for structured data, such as a split operator, first-class functions, iteration, and dynamic window boundaries. Chimera is parsed into an abstract syntax tree and then translated to core relational algebra, from which Bro event code is generated.

Borders demonstrated Chimera code for detecting side-jacking attacks, DNS TTL value changes, DNS tunnels, and phishing scenarios. To evaluate the potential loss of efficiency in automatically converting Chimera code to non-optimized Bro code, Borders compared the efficiency of handwritten Bro code and Chimera code for a number of different events, finding only about a 1% speedup for handwritten and optimized code. He hoped that the increased usability in allowing many more people to use a standard, declarative language rather than writing procedural code for network data analysis would outweigh these small inefficiencies.

Christian Kreibich (ICSI) explained that he finds it useful to use attributes on tables to manage state in the Bro IDS, and he wondered what sorts of these types of tunings were exposed to users. Borders noted that each table has a maximum limit, but that the limit is user-accessible. Ian Goldberg (University of Waterloo) questioned whether Chimera aims to handle adversaries who attempt to evade the underlying IDS. However, Borders said these kinds of concerns were orthogonal to Chimera.

New Attacks on Timing-Based Network Flow Watermarks

Zi Lin and Nicholas Hopper, University of Minnesota

Watermarking a network flow is the process of applying a scheme to packets as they traverse a network so that this same sequence of packets can be detected at the other end of the network regardless of tunneling, re-encryption, or the use of anonymity networks. For instance, statistical changes in timing delays for a flow can be used for watermarking. Recent work has focused on making watermarking schemes that are transparent to a passive observer since an adversary can destroy a detected timing-based watermark through the addition of his or her own delays.

In this presentation, Nicholas Hopper (University of Minnesota) argued that current efforts to define schemes as secure against a passive observer are insufficient for providing security guarantees. He suggested that passive invisibility as

an attack model should be replaced by a chosen flow attack, analogous to a chosen plaintext attack in cryptography. In this model, a "warden" would know and could even choose the distribution of traffic for a network flow and would then be tasked with detecting whether that flow had been marked or unmarked during its travel. Hopper showed how this model illuminates an attack against the Rainbow scheme (NDSS '09) in which the warden knows the interpacket delays at the beginning of the flow and can then create a histogram of interpacket delays on the receiver side; the histogram of a marked flow would no longer be centered around 0. A similar line of reasoning in this model also showed weaknesses in the Swirl scheme (NDSS '11).

Ian Goldberg (University of Waterloo) questioned where in the attack against the Rainbow scheme the warden selected the attack. Hopper clarified that the warden need not choose the flow, but rather only needed to know which flow.

Invited Talk

Summarized by Benjamin Braun (bjmnbraun@gmail.com)

Nature Is a Language; Can't You Read?

Meredith L. Patterson, DIYBiologist and Senior Research Scientist, Red Lambda

Meredith Patterson began by illustrating the similarity between the homebrew computer community of the 1970s and the DIYbio community. Modern synthetic biology research labs must invest large sums in their machinery, which includes cell incubators, centrifuges, autoclaves, and more. Patterson compares such labs to mainframe computers, and proposes that DIYbio can reduce the cost of performing synthetic biology, just as the personal computer has done for computing.

Patterson then provided an introduction to the concepts, terminology, and techniques of synthetic biology from the perspective of a computer scientist. The introduction included a practical overview of how to induce cell competence, manipulate DNA, run gel electrophoresis, and much more. Interestingly, the cell and a computer program are strikingly similar. By analogy, the goal of synthetic biology is to program cells for some general purpose function.

Synthetic biology research is driven by commercial demand to efficiently create certain drugs and biofuels and the demand for new drugs. Artemisinin, an antimalarial drug found in wormwood plants, can be efficiently produced using engineered cells. Various kinds of biofuel-producing bacteria have been created. The chimeric monoclonal antibody Infliximab, a treatment for certain autoimmune disorders, was developed using the techniques of synthetic biology.

The second half of her talk listed a large number of tools have been successfully developed for DIYbio. Highlights included the DremelFuge, an attachment-for-Dremel tool which spins microcentrifuge tubes at over 25,000 Gs; the use of insulin syringes as a replacement for micropipettes; and an electroporator made from an electric lighter.

Patterson ended with predictions of how synthetic biology will change in the near future. Creating a strain of bacteria programmed to perform a specialized task can take years of effort, even for the best-equipped labs. The construction of something like an FPGA made from microfluidic channels, where a high level description of a specialized task can be converted into a model organism, would accelerate the field of synthetic biology. DIYbiologists may be the first to create such a general purpose system for synthetic biology.

Perry E. Metzger pointed out the lack of a unified higher level language for synthetic biology. Patterson responded that even the MIT biobricks project, which provides higher level building blocks for synthetic biology, is not DIYbiology friendly, because it favors E. coli bacteria and is restricted to academic use. Patterson predicts that higher level languages for specifying novel cellular functions will continue to be developed over time. Someone was interested in how intellectual property and copyleft relates to DIYbio. Patterson pointed out that many of the devices shown in the talk have open-source blueprints. She also hopes that the DIYbio community will start to produce patentleft work. Someone was curious whether there was any overlap between the DIYbio group and the alcohol homebrew community. Patterson responded that a Rice University IGEM team had created a strain of brewer's yeast that produced resveratrol during the fermentation process. She had also heard of GFP (green fluorescent protein) beer. Patterson recommends that those wanting to get their start in DIYbio visit diybio.org, www.openwetware.org, or come to the open science summit (opensciencesummit.com).

Poster Session

Summarized by Srinath Setty (srinath@utexas.edu)

User Interface Toolkit Mechanisms for Securing Interface Elements

Franziska Roesner, James Fogarty, and Tadayoshi Kohno from University of Washington.

Franziska Roesner presented this poster. In user interface toolkit research, it is common to find an assumption that a single developer owns the entire interface. But this assumption isn't true in the world of mashups. For instance, an Android application embeds application UI elements as well as advertisements; in the case of Web applications, a single Web page may have elements from multiple providers.

As a more concrete example, consider an Android application that embeds an advertisement. In such a scenario, the application developer must trust, i.e., assume, that the advertisement code doesn't incorrectly use some of the privileges

that user may have granted to the application. A traditional approach to solve this problem is to explicitly ask permissions from the user when one component in an application tries to access another component or use permissions of another component. But this approach may not be user friendly in many cases.

The main goal of this poster is to motivate the need to consider security as one of the primary goals in user interface toolkit research. The authors examine existing approaches to solve this problem and propose mechanisms to isolate components from mutually distrusting providers. The authors have also implemented a toolkit for Android and a browser-based toolkit.

Verification with Small and Short Worlds

Rohit Sinha and Cynthia Sturton, University of California, Berkeley; Petros Maniatis, Intel Labs; Sanjit A. Seshia, and David Wagner, University of California, Berkeley

Cynthia Sturton presented this poster. The motivation behind this work is that hypervisors and CPU emulators are used in a large number of security-critical applications. Examples include cloud computing, malware analysis, and hosting dangerous applications. One of the ways to discover security critical bugs in these large software systems is to verify their safety properties via finite-state model checking. However, one of the major limitations of existing verification mechanisms is that they don't scale to large problems. More specifically they cannot be used to verify safety properties of large data structures.

Verifying isolation properties of hypervisors involves verifying safety properties of large data structures that the hypervisor uses. Example data structures include page tables, translation lookaside buffers (TLBs), and caches. To solve this problem, the authors present a new technique called S2W (Small and Short Worlds) for verifying the safety properties of large and unbounded data structures.

S2W includes three steps: (1) standard mathematical induction, (2) small world, and (3) short world. In the first step, if the property that the system is verifying is an invariant, the system returns true, else it moves to step two. In the second step, the system creates a scaled-down model with many of the data structure components abstracted. This is one of the key mechanisms that makes the state exploration feasible. In the third step, using the bounded model of the data structure, model checking is used to prove the invariant on the abstracted system. Evaluation shows that this system is practical, and the authors used their system to verify safety properties of six software systems (Bochs' TLB, Content-addressable memory-based cache, Shadow page tables, SecVisor, sHype, and ShadowVisor).

Exploiting URL Shortening Services for Botnet C&C

Cui Xiang, Liao Peng, and Jin Shuyuan, Wang Shuai Institute of Computing Technology, Chinese Academy of Sciences, China

Cui Xiang presented this poster on a system called Flux. Flux is a robust and efficient botnet command and control (C&C) protocol. Flux can support large-scale management of botnets and is secure against DNS redirection. Flux uses a URL shortening service together with cloud-based file hosting to achieve its properties. The proposed mechanism has the following advantages: (1) a botmaster could upload a resource (e.g., JPEG file, MP3 file) which embeds encrypted and signed commands to cloud-based file hosting services (Flux requires publicly accessible servers with static IP addresses); (2) irrespective of the URL used for the resource, URL shortening services are used in Flux to subsequently locate the resource. More specifically, URL shortening services provide a persistent mapping between long URLs to short ones. Moreover, some of the URL shortening services provide custom aliases for long URLs.

The complete protocol can be divided into four steps. First, a botmaster encrypts and signs the commands to be executed; the botmaster then embeds the ciphertext into a small resource file (JPEG, MP3 etc.) and uploads the resource file into a file hosting service. The file hosting service returns a long URL that will allow anyone with the URL to download the resource file. Second, the botmaster randomly selects a username generated by a hardcoded username generation algorithm. Third, a bot enumerates the predefined URL shortening services one by one using the username generation algorithms. Fourth, the bot downloads the resource and verifies the signature before executing the command embedded in the resource file.

The authors also presented countermeasures. First, after the C&C analysis procedure, the defenders could deploy an infiltrator to track the activities inside the botnet. Second, the defenders should focus on the potential abuse of URL shortening services and cloud-based file hosting services.

Kernel-Level Sandbox of Application Processes with High Assurance

Hussain M. J. Almohri Danfeng (Daphne), Yao Dennis Kafura, Virginia Polytechnic Institute and State University

The authors proposed mechanisms to build a kernel-level sandbox for applications that require high assurance. The technical challenges include (1) designing a mechanism that provides a strong binding of processes to trusted applications at runtime, (2) ensuring that the overhead of cryptographic operations on the kernel's performance is minimal, and (3) detecting malicious interactions with the kernel.

The threat model of the authors assumes that the adversary has no physical access to the hardware, the kernel, the kernel

interface, and the mandatory access control layer; that legitimate applications are vulnerable; and that malware runs as a stand-alone process. The approach that the authors take is to use lightweight cryptography for authentication and isolation of malware at runtime. Also, the kernel-level sandbox includes two mechanisms: (1) associate symmetric keys that are known only to the kernel and the application that owns the symmetric keys (this mechanism allows the kernel to uniquely identify applications and prevent forging of process identities), and (2) decouple authorizing system calls from authenticating applications.

Performance evaluation shows that the individual system call overhead is around 3x that of the baseline, and the end-to-end overhead of sandboxing processes is within 26% of a system that doesn't include the sandbox. Finally, the authors conclude that their mechanisms provide strong and unforgeable application identities and that the malicious applications are completely isolated with minimal overhead on the end-to-end performance.

Computer Security and the Modern Home

Tamara Denning, Tadayoshi Kohno, and Henry M. Levy, University of Washington

The goal of this work is to look at computer security for the home technology ecosystem. The motivation for this work is that security will be more important in the home ecosystem when a lot of devices are used at home in the near future. The authors list many factors by which home technology ecosystems differ from standard computer security: they (1) deal with a number of human assets, (2) contain increased sensor and actuator capabilities, (3) are not professionally managed, (4) use diverse technologies, (5) have diverse stakeholders, and (6) can produce mismatched expectations. The authors also analyzed the security of a mobile Webcam toy, a wireless scale, and a home security siren. The details of this poster will be in an upcoming Communications of ACM article.

Using Interactive Static Analysis for Early Detection of Software Vulnerabilities

Jun Zhu, Jing Xie, Bill Chu, and Heather Lipford, University of North Carolina at Charlotte

The authors motivated their work by making a simple observation: most of the security vulnerabilities occur because of developers who have no experience with writing security-critical code. Existing solutions include static analysis (an analysis that doesn't require executing the program) and dynamic analysis (an analysis that requires symbolically executing the program). One of the main drawbacks of these approaches is that they are done after the program or software is completely developed. As a result, vulnerabilities are not detected early in the software development cycle. The authors proposed a mechanism that uses interactive static analysis to detect software vulnerabilities as and when the software is developed.

The authors' mechanism includes two analyses: (1) interactive data flow analysis to assist the developer to handle untrusted data properly and (2) interactive control flow analysis; the programmer will be assisted with writing code that respects security policies (for instance, access control policy). The authors performed an extensive user study of their mechanism. For the data flow analysis, they found that 69% of the warnings were clicked on and 49% of the warnings were resolved. For the control flow analysis, they found that the tool detected many zero-day vulnerabilities in Apache Roller 5.0 (an open source blog server) and these vulnerabilities were later confirmed by penetration testing. The tool also found seven cross-site request forgeries in Apache Roller. The authors conclude that their mechanisms find real security bugs not detected by existing mechanisms and that their tool can help software developers detect vulnerabilities early in the software development cycle.

Empirical Evaluation and Pushback of Malicious Web Advertisements

Robin E. Gonzalez V. and Michael E. Locasto, University of Calgary

The motivation for this work is the rise of Web advertising: the concept of reselling ad space empowers malicious actors on the Web to inject malicious advertisements into popular Web pages.

One of the main goals of this work is to build a tool to help collect data about the following four measurement tasks associated with malicious Web advertisements: (1) find the relationship between advertisements on Web sites and malware detection, (2) determine how often advertisements change on Web sites, (3) determine the advantages of employing a pushback mechanism, and (4) analyze how the top 500 Web sites compare with respect to advertisement vs. malware detection.

The authors built a tool to measure the malicious content injected by reselling ad spaces. The tool is a Firefox extension and employs a sampling strategy. The tool automatically probes the top 500 Web sites and profiles the advertisements displayed in those sites using a set of publicly available malware detectors. The authors also built a pushback mechanism to notify the Web servers that have participated in delivering malicious advertisements.

The Rise of the App-Net

Md Sazzadur Rahman, Ting-Kai Huang, Harsha V. Madhyastha, and Michalis Faloutsos, University of California, Riverside

The goal of this work is to develop a better understanding of the ecosystem of malicious applications in Facebook. The motivation for this work is the presence of a large number of

malicious applications on online platforms like Facebook. For example, the authors examined 111,000 applications and found that 13% of them are malicious; the authors also report that 60% of the malicious applications get at least 100,000 clicks. Moreover, 40% of the malicious applications have a median 1000 monthly active users, according to the authors' study.

The authors created an application on the Facebook platform called MyPageKeeper to study malicious applications on Facebook. Their application identified 6000+ malicious applications. Based on their study, the authors make several interesting observations: (1) malicious applications promote each other, (2) malicious applications extensively collaborate, (3) malicious applications have a high local clustering coefficient (local clustering coefficient for a node is "the number of edges among the neighbors of a node over the maximum possible number of edges among those nodes"), (4) colluding applications exhibit name similarity among themselves, and (5) indirection. Web sites are often hosted on Amazon AWS (for example, 84 out of 103 indirection Web sites were shortened by bit.ly and one-third of them resolved to Amazon's AWS).

Classification of UDP Traffic for DDoS Detection

Alexandru G. Bardas, Loai Zomlot, Sathya Chandran Sundaramurthy, and Xinming Ou, Kansas State University; S. Raj Rajagopalan, HP Labs; Marc R. Eisenbarth, HP TippingPoint

The motivation for this work is the increase in denial of service attacks. The authors quote an important observation by Gill et al. to motivate their proposed solution: "During normal operation, the packet rate of traffic going to an address is proportional to the packet rate of traffic going from that address." The authors hypothesize that under normal operation, the ratio between traffic from the source to destination to the traffic from the destination to source will be less than a pre-defined maximum threshold value. The authors also hypothesize that this value can be used to separate normal traffic from malicious traffic.

Based on their testbed analysis, the authors made several interesting observations: (1) benign applications use UDP traffic to maintain constant communication between sender and receiver (e.g., NFS); (2) in many benign cases, there is an initial communication followed by a one-way burst of UDP packets (e.g., SIP); and (3) in many applications, there is a one-way burst of UDP traffic (e.g., older versions of NetFlow).

A DCF-Based Covert Timing Channel for IEEE 802.11 With Off-the-Shelf Wireless Cards

Sakthi V. Radhakrishnan, A. Selcuk Uluagac, and Raheem A. Beyah, Georgia Institute of Technology

A covert communication channel can be used to hide secret messages within regular traffic. The authors observed that wireless networks are promising as covert channels, specifically, the networks that employ multiple access with collision avoidance (CSMA/CA), including 802.11 networks. This is because these networks introduce randomness into the network and hence provide a good cover for the covert channel. The authors' mechanism exploits the random backoff in the distributed coordination function (DCF) to realize a relatively high-bandwidth covert timing channel.

The authors' implementation uses off-the-shelf wireless cards: NetGear router with ZyXEL access point, Cisco Aironet WiFi cards, and Qualcomm Atheros PCMCIA cards. Finally, the measurement shows that the covert channel bandwidth is 1142 packets per second.

Program Structure-Based Feature Selection for Android Malware Analysis

Andrew Walenstein, Luke Deshotels, and Arun Lakhotia, University of Louisiana

The problem that the authors look at is that the Android malware mimics malicious code with benign code found somewhere else in many cases. Examples include infected and repackaged applications. Benign code poses challenges for classification and clustering since it introduces errors. The challenge is in selecting a set of features for clustering and classification. The authors exploit the idea that a malicious application repackaged in a benign application will be programmatically independent, allowing them to select a set of features and perform clustering and classification. The early results indicate that the proposed feature selection treatment generates an ideal clustering solution. In future work, the authors plan to look at analyzing the entire Android malware Genome project data set.

Dismantling iClass and iClass Elite

Flavio D. Garcia, Gerhard de Koning, and Gans Roel Verdult, Radboud University Nijmegen, The Netherlands; Milosch Meriac, Bitmanufaktur GmbH

The authors examine the security of the ISO/IEC 15693 14443-B compatible smartcard that uses proprietary cryptography with a 64-bit key. This was introduced in 2002 as a replacement to HID Prox; the product was rebranded as PicoPass manufactured by Inside Secure. According to HID, 300 million cards were sold and there are two versions: iClass Standard and iClass Elite. iClass Standard uses one master key for every system worldwide; iClass Elite allows customizing the master key and is also more expensive. The authors reverse engineered the authentication protocol and recovered the master key. Recovering the master key on iClass Standard took less than a day on a laptop, and it was many orders of magnitude faster with iClass Elite.

Poster Session

Summarized by Benjamin Braun (bjmnbraun@gmail.com)

FlexCOS: Enabling Academic Smartcard Research

Kristian Beilke and Volker Roth, Freie Universität Berlin

Kristian Beilke (kbeilke@zedat.fu-berlin.de) presented a model smartcard built using an FPGA board, FlexCOS. The smartcard industry keeps its designs and software behind closed doors, restricting academic research of smartcards. Beilke hopes that FlexCOS will lead to an increase in security research on smartcards.

Transparent Probabilistic Packet Marking: Proposal and Development

Akira Kanaoka and Nasato Goto, University of Tsukuba; Masayuki Okada, Japan Network Information Center; Eiji Okamoto, University of Tsukuba

Akira Kanaoka (kanaoka@risk.tsukaba.ac.jp) presented an IP traceback method based on probabilistic packet marking (PPM) which requires fewer packets to rebuild a traceback than existing methods. Installing transparent packet-marking routers in a network further increases the efficacy of their IP traceback method. Kanaoka and his coworkers have transformed various kinds of routers into transparent PPM devices.

Tracing Attacks on Advanced Persistent Threat in Networked Systems

Masahiko Kato, University of Tsukuba/Internet Initiative Japan Inc.; Takumi Matsunami, Kyushu Institute of Technology; Akira Kanaoka, University of Tsukuba; Hiroshi Koide, Kyushu Institute of Technology/Information-Technology Promotion Agency, Japan; Eiji Okamoto, University of Tsukuba

The researchers created a graph model of computer networks, based on a simplified classification of network devices. This model allows for the study of advanced persistent threat (APT) in a simulated system and for the evaluation of efficient defense strategies against APT. The researchers are working on generating such models from existing networks using automated methods. A GUI for viewing the model and simulating APT is also in the works.

A Security Aware Stream Data Processing Scheme on the Cloud

Katsuhiro Tomiyama, Hideyuki Kawashima, and Hiroyuki Kitagawa, University of Tsukuba

Katsuhiro Tomiyama presented a built secure scheme for storing data streams such as financial information or network packets on the public cloud. In the scheme, a sensor generates data tuples and sends them to the cloud storage provider in three encrypted forms. A client sends queries to the cloud storage, which executes the queries on the encrypted data and then sends the client the query response in the form of encrypted tuples. Tomiyama describes two refinements which increase the throughput of the scheme.

Secure Out-of-Band Remote Management in Infrastructure as a Service

Tomohisa Egawa, Naoki Nishimura, and Kenichi Kourai, Kyushu Institute of Technology

The researchers present a built scheme, FBCrypt, which protects a user VM from attacks originating from the management VM in infrastructure as a service. In the scheme, encryption protects the inputs generated by the VNC client and the framebuffer output by the remote user VM from being accessed by the management VM. The VMM, whose integrity is guaranteed using remote attestation, performs encryption and decryption on behalf of the user VM. The researchers evaluate the increase in keyboard response time and full screen update response time caused by FBCrypt.

Telerobotic Surgery Meets Information Security

Tamara Bonaci and Howard Jay Chizeck, University of Washington

Tamara Bonaci (tbonaci@uw.edu) studies security problems in telerobotic surgery systems, which allow surgeons to operate on a patient in a remote area or war zone from miles away. Disturbing security vulnerabilities have been discovered in telerobotic surgery systems. These vulnerabilities motivated analyzing the full designs of telerobotic surgery systems from a security perspective.

Virtualizing Secret-Shared Database System

Yutaka Nishiwaki and Hiroshi Yoshiura, University of Electro-Communications, Tokyo, Japan

Databases using the cryptographic technique of secret sharing can be used to securely store and query sensitive data. The researchers describe how secret-shared databases can implement database virtualization. Secure indices allow secret-shared databases to more quickly retrieve queried data without sacrificing security.

Identifying Anonymous Posts of Job Seekers

Tomotaka Okuno and Hiroshi Yoshiura, University of Electro-Communications, Tokyo, Japan

Tomotaka Okuno presented a method for measuring the similarity between an anonymous user's twitter feeds and a job resume. This allows nosy employers to determine which twitter user, who may have embarrassing tweets on a twitter feed under a different name, corresponds to the job applicant. Given 10 possible twitter feeds and a job resume, the method successfully determines which feed is run by the applicant.

MalCut: Malware-Initiated Data Leakage Prevention System

Deok Jin Kim, Byung Jin Han, Young Han Choi, and Byung Chul Bae, The Attached Institute of ETRI, Republic of Korea

The researchers find that many malware applications do not fill out the HTTP referrer field correctly. They propose a system, MalCut, which detects HTTP requests where the referrer field has not been correctly filled out.

Aiding Malware Detection Using Large-Scale Machine Learning

Yazan Boshmaf, Matei Ripeanu, and Konstantin Beznosov, University of British Columbia; Kyle Zeeuwen, David Cornell, and Dmitry Samosseiko, Sophos Labs

Yazan Boshmaf (boshmaf@ece.ubc.ca) presented AUGUR, a rapid malware detection system. AUGUR generates software signatures which can identify suspicious software, allowing a faster response to the victim of a new malware attack. The signatures are generated using a large scale machine learning algorithm learning a variety of classifiers, including support vector machines and decision trees.

Control-Alt-Hack: A Card Game for Computer Security Outreach, Education, and Fun

Tamara Denning and Tadayoshi Kohno, University of Washington; Adam Shostack, Microsoft

Tamara Denning (tdenning@cs.washington.edu) presented the educational game Control-Alt-Hack. The game is intended to sharpen players' awareness of the impact of computer security in their lives. Players take control of white hat hackers, and must complete hacking missions to win. The designers plan to sell the game at an online retailer, and educational copies are also available.

Web Security

Summarized by Alexandros Kapravelos (kapravel@cs.ucsb.edu)

On Breaking SAML: Be Whoever You Want to Be

Juraj Somorovsky, Ruhr-University Bochum; Andreas Mayer, Adolf Würth GmbH & Co. KG; Jörg Schwenk, Marco Kampmann, and Meiko Jensen, Ruhr-University Bochum

Juraj Somorovsky presented an extensive evaluation of the security properties in frameworks that use SAML, an XML-based language designed for making security statements about subjects. SAML is an OpenID alternative that solves the Web browser single sign-on problem. This critical component was found to be vulnerable to XML Signature Wrapping (XSW) attacks, providing a single point of failure for sign-on systems. The attacker can bypass the checks of the system by wrapping a valid XML signature into another SAML object.

An extensive evaluation of 14 SAML frameworks and systems has been performed, and the authors found 11 of these frameworks suffer from XSW vulnerabilities. One additional framework was susceptible to a more sophisticated XSW attack, and three of the frameworks were also vulnerable to Signature Exclusion attacks. In total only two frameworks were resistant to all test cases, which reveals that XSW vulnerabilities are a real threat in practice.

To overcome this problem the authors implemented the first fully automated penetration test tool for XSW attacks in SAML-based frameworks and integrated it into the WS-Attacker framework. The tool supports several attack vectors, including XML Schema validation, placement of the Signature element, and Signature exclusion. Countermeasures are also proposed: only process what is hashed, ignoring everything else, and mark signed elements instead of just returning a Boolean value whether the document is signed or not. The authors have also notified the responsible security teams of the vulnerable frameworks, and in many cases the issues were fixed.

Clickjacking: Attacks and Defenses

Lin-Shung Huang, Carnegie Mellon University; Alex Moshchuk, Helen J. Wang, and Stuart Schechter, Microsoft Research; Collin Jackson, Carnegie Mellon University

Lin-Shung Huang presented an extensive study on clickjacking, an attack that creates the misconception in the user that she is interacting with one principal while in reality another UI element is triggered. Lin-Shung went through some existing attacks and defenses and showed that those are insufficient against a determined attacker.

In addition to existing attacks, three new attack variants were proposed. The cursor-spoofing attack tricks the user by presenting a fake mouse cursor and hiding the real cursor, which can make the user click the wrong button. Another attack was the double-click attack that asks the user to double-click on a button, but after the first click an aligned pop-up window presents a new button at the same place, tricking the user this way to click a button on the pop-up window. The last attack proposed is based on the whack-a-mole game, where the user is clicking on a button that moves with a fake cursor as part of a game and at some point a sensitive button appears under the real cursor, causing the victim to click it by accident.

These attacks motivated the authors to create a new defense mechanism: InContext defense. The key observation is that a sensitive UI element is presented to the user out of context, and the user operates with wrong assumptions. Ensuring the context integrity at the OS or browser level is the fundamental contribution of the proposed defense. The authors' solution is based on ensuring the visual and temporal integrity of sensitive user actions. For example, they disable cursor customization when sensitive elements are present or they delay sensitive elements until they are fully visible for a certain amount of time. To evaluate their attacks and defenses they used Amazon's Mechanical Turk and showed how effective their defenses are in practice by using real users.

Privilege Separation in HTML5 Applications

Devdatta Akhawe, Prateek Saxena, and Dawn Song, University of California, Berkeley

Devdatta Akhawe introduced a new design for effective privilege separation in HTML5 applications. The current model is not enough, since privilege separation is based on the same

origin policy, which means that to separate different application components we need to host them to different Web origins. In more recent application platforms, like the Google Chrome extension platform, the situation is slightly better: install-time manifest files declare the components that use privileged APIs. Still, the authors found two fundamental problems with this approach: bundling (multiple components running in the same principal) and TCB inflation (how much more code has higher privileges than the application core that actually needs them).

The proposed architecture design is different from other approaches since it maintains compatibility and facilitates adoption. It is based on a hierarchical structure, where there is only one privileged parent component and all children are unprivileged and isolated: each one is run in its own temporary origin. The role of the parent is to guard access to the powerful API provided by the platform. In order for the children to make privileged calls, they communicate with the parent, who according to an application-specific policy allows or blocks the request.

The authors have implemented their design and used it on three different case studies and on the top 50 Google Chrome extensions. For all three case studies they changed only 24 lines of code but managed to reduce the TCB dramatically, which also shows the limitations of current platforms' privilege separation. Moreover, the performance overhead is negligible in terms of additional load time and memory consumption.

Software Security I

Summarized by Srinath Setty (srinath@utexas.edu)

Fuzzing with Code Fragments

Christian Holler, Mozilla Corporation; Kim Herzig and Andreas Zeller, Saarland University

Christian Holler explained that fuzzing is a software engineering technique used to expose vulnerabilities in software during the testing process. At a high level, fuzzing involves providing random inputs to the software being tested and checking the output of the software to see if the software crashed. Fuzzing is common in software engineering, especially for exposing security vulnerabilities. One of the main advantages of fuzzing is that the approach can be used on multiple versions of the software to check if the new version behaves differently when compared to the old version on the same inputs. Fuzzing can also be used for checking the correctness of programs.

There are lots of tools available (both academic and commercial), but the focus of this paper is to expose vulnerabilities in the JavaScript interpreter in modern browsers. Fuzzing a JavaScript interpreter is an important problem because a JavaScript interpreter usually executes third-party, untrusted scripts fetched from a network. A JavaScript interpreter usually contains two components: a parser and a runtime. Inputs are first seen by the parser, and only the semantically valid inputs are passed to the runtime. So if fuzzing is used to expose vulnerabilities in an interpreter, the fuzzing software must make sure that the inputs generated are semantically valid and are seen by the runtime.

One possible approach is to write a fuzzer that is tailored to a language like JavaScript, but the approach is not scalable and is also a high-maintenance task (since a new fuzzer needs to be written for every new language). The goals set forth by the authors were: (1) create a fuzzer that is generic (i.e., works for all interpreted languages); (2) make only general assumptions about languages; (3) find real-world defects.

The authors built a tool called LangFuzz which consists of three steps. First, the tool uses sample code to learn the underlying grammar of the language. Second, the tool generates mutated test cases; the tool randomly picks a code fragment or it uses the learned grammar to generate a syntactically valid test case. Third, the tool feeds the test case into the interpreter and checks for crashes and assertion failures. A good thing about the tool is that it locates incomplete fixes and finds new defects.

To evaluate LangFuzz, the authors compared their tool to the state of the art, jsfunfuzz, a tool developed by Mozilla in 2007 specifically for JavaScript. It has found over 1000 bugs and is highly specialized for JavaScript. The authors compared the number of defects found by both their tool and jsfunfuzz: jsfunfuzz found 15 bugs and LangFuzz found eight bugs of which only three overlapped with the bugs found by jsfunfuzz. The authors also did a field test within Mozilla for about four months. They found 51 bugs of which 20 were security-critical bugs.

The authors also evaluated their tool with a different language: PHP. The tool found 18 bugs in two weeks with PHP. The authors think that the tool will produce poor results with statically typed languages like C++ or Java. However, if enough code fragments are provided, the tool can be used with C++ and Java. But, as of now, the authors recommend using their tool with weakly typed languages.

Lior Malka commented that the code that goes through the runtime stage can still have bugs. Holler replied that it is true and if there is a second implementation, it is possible to catch correctness errors. Someone else asked whether there are existing tools for doing that. Holler replied that there are probably no language-independent tools, but jsfunfuzz can do correctness checks. Mihai Christodorescu asked if the tool can work with macro languages (e.g., LaTeX). Holler replied no since the tool requires the syntax to be static.

kGuard: Lightweight Kernel Protection against Return-to-User Attacks

Vasileios P. Kemerlis, Georgios Portokalidis, and Angelos D. Keromytis, Columbia University

Vasileios Kemerlis started by pointing out that Linux alone had more than 140 assigned CVE numbers in 2010 out of which 12 were privilege escalation attacks and 13 were bugs that could be triggered remotely (e.g., kernel memory leaks, authentication bypass, and denial of service). Kernel attacks are becoming more common because they are a high-value asset and have a large attack surface, but exploiting privileged-user space processes has become harder because of increased use of defenses like canaries, address space randomization, etc. Recall that the return to libc attacks corrupt the kernel memory and inject code or reuse existing code to mount an attack.

The focus of this work is to look at return-to-user attacks. Kemerlis described kGuard, a lightweight solution that augments kernel code with assertions about control flow. As a result, privileged execution is sandboxed within the kernel and doesn't run into user space code. At a very high level, kGuard makes a realistic threat ineffective. Moreover, kGuard introduces a code diversification technique that prevents attacks against kGuard.

At a very high level, kGuard places a code fragment before every exploitable control transfer. Then later, it verifies that the target address of an indirect branch is always inside the kernel space. If the assertion is true, execution continues; otherwise, control is transferred to a runtime violation handler. Note that to attack kGuard, an attacker would first have to find the address of an indirect control transfer instruction inside the kernel. Also, the attacker should be able to fully control the value of the target address. The authors observed that such an attacker can more easily elevate his privileges by overwriting the credentials associated with a process under his control.

kGuard also implements two diversification techniques to defend against bypassing attacks. The first technique is called "code inflation," and it reshapes the kernel's text area (i.e., the location of every control flow instruction is at a random location). The second technique is called "CFA motion," and it continuously relocates the protected branches and injected guard code fragments.

To evaluate kGuard, the authors instrumented 10 different vanilla Linux kernels. The authors tested eight exploits. In every case, kGuard could detect and prevent the attack. The authors also measured its performance. The slowdown was less than 1% in real-life apps and was about 11.4% in the case of lmbench.

Lior Malka (Intel) mentioned that Intel and ARM processors have a protection for jumps from ring 0 to ring 3; why didn't the authors use those features? Kemerlis replied by saying SMEP is the feature. Malka mentioned other features: with Intel's processors, it is possible to mark certain pages as user and certain pages as supervisor and, hence, to protect control flow from supervisor to user. Kemerlis pointed out that the approach would require rewriting the operating system; the approach proposed in the paper doesn't require any rewriting and it is automatic. Rik Farrow from USENIX asked whether the authors tried tuning and how important was the length of the sled. Kemerlis said that they didn't run extensive benchmarks to find the effect of sled length. Kemerlis also mentioned that 5% space overhead isn't bad. Vishwanath Mohan from UT Dallas asked if kGuard instruments return statements. The answer was yes.

Enhanced Operating System Security Through Efficient and Fine-Grained Address Space Randomization

Cristiano Giuffrida, Anton Kuijsten, and Andrew S. Tanenbaum, Vrije Universiteit Amsterdam

Cristiano Giuffrida noted that kernel exploitation is gaining momentum. There are many exploits for Windows, Linux, etc. Moreover, there are many memory error vulnerabilities. In fact, there are many attack opportunities for both local and remote attacks.

Existing defenses for these attacks include preserving kernel code integrity, kernel hook protection, and control flow integrity. However, there is no comprehensive memory error protection, required virtualization, or address space randomization (ASR) for operating systems. Only the recent versions of the Windows operating system performs a basic text randomization. The main challenges with fine-grained address space randomization for operating systems includes the following. First, instrumentation is needed for fine-grained ASR and may impose high runtime overheads. Moreover heavy instrumentation introduces a lot of untrusted code that may impact the reliability of the operating system. Second, there are more avenues for leakage of information. Third, an ASR solution should be resilient to brute-forcing attacks to which many of the recent proposals for ASR have been vulnerable . Fourth, an ASR solution should rerandomize the address space to reduce successful attacks over time.

The authors built a system where the layout of memory objects is unpredictable. The authors used the LLVM-based link time transformations for safe and efficient ASR. As a result, a minimal amount of untrusted code is exposed to the runtime. Their solution also supports live rerandomization to maximize unobservability of the system. Moreover, there are no changes in the software distribution model.

The system performs many transformations to implement fine-grained ASR. First, code is randomized by randomly permuting all program functions; in LLVM, this is possible by permuting the symbol table. Second, static data is randomized to randomly permute all the static data and read-only data in the symbol table; other transformations include employing a random padding strategy with dummy variables; also, buffer variables are separated from other variables to reduce the impact of buffer overflow attacks. Third, the base address of the stack is randomized and so is the relative distance between any two objects. Finally, dynamic data is randomized by randomizing the start address and by adding random-sized padding between different objects.

Moreover, the system also supports live rerandomization which is the first stateful live rerandomization technique to periodically rerandomize kernel state. It supports arbitrary memory layout changes, and the system also sandboxes the rerandomization code to recover from runtime errors.

The authors measured the overheads with ASR instrumentation. In most cases, the performance is within 5% of a system that doesn't provide ASR and is about 35% in the case of perlbench because it's a memory-intensive benchmark. The authors also measured runtime overhead when using different rerandomization latencies. They observed that the overhead increases exponentially.

Rik Farrow (USENIX) asked if the authors tested their system against kernel exploits to tune the rerandomization latency. Giuffrida answered that it was not done. He also mentioned that it is not easy to tune generally and that it may be okay to give up some performance depending on the vulnerability. Deskin Miller (Microsoft) asked if the defense would work if the attacker can manipulate some property of memory layout. Giuffrida answered that if the system tries to randomize the corrupted state, it may make it worse since tainted state may lead to worse situations. Michael Franz (UC Irvine) asked if the rerandomization is an atomic operation and would that be a bottleneck on a multicore system. Giuffrida answered that it is ongoing work; moreover, in the team's OS architecture, each OS component will have different replicas, and if one component is randomized, there will be another component available to reduce unavailability by multiplexing. Also, the rerandomization takes only a short time in many cases when the OS components don't have a lot of state. Franz asked whether the system could do anything to improve availability. The system can rollback live rerandomization at any time.

Botnets and Web Security

Summarized by Gianluca Stringhini (gianluca@cs.ucsb.edu)

From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware

Manos Antonakakis, Damballa Inc. and Georgia Institute of Technology; Roberto Perdisci, University of Georgia and Georgia Institute of Technology; Yacin Nadji, Georgia Institute of Technology; Nikolaos Vasiloglou and Saeed Abu-Nimeh, Damballa Inc.; Wenke Lee and David Dagon, Georgia Institute of Technology

Manos Antonakakis presented this new approach to fight botnets. The idea is to detect domain generation algorithm (DGA) malware without knowing the DGA algorithm itself.

In his previous research, Manos showed how limiting DNS abuse helps limit the abuse on the Internet. Unfortunately, current techniques cannot act proactively at the recursive level of DNS. For example, the previous systems called NOTOS requires a long tail of DNS requests before making a decision. The approach they propose is to perform early malware domain detection by detecting rising DGA botnets. To do this, they leverage the observation that DGA bots are likely to generate a large number of NXDOMAIN responses when trying to contact the command and control server using not yet registered domains. Moreover, such domains are likely to share similar properties (e.g., the domain name length), since they are generated by the same algorithm.

In this spirit, Manos explained how they group and classify NXDOMAIN responses to detect groups of requests likely generated by the same bot. As a practical approach, one could detect malware-infected machines, and then look at where they connect to detect command and control servers.

For their evaluation, they analyzed 15 months of data, collected from ISP traffic (Damballa customers), and showed that they could correctly group and classify DGA-generated queries. Manos then showed as a case study the BankPatch botnet. This botnet targets 187 banks, and steals credentials any time a victim tries to connect to one of them. The DGA used by the botnet was composed of four random characters and by a domain argument. He showed how 270 different networks contained infected machines. The command and control servers were located in Eastern Europe.

Of course, this system has limitations. First of all, sometimes their HMM model (which is used as one of their classification features) fails. This is an implementation problem. Second, they cannot attribute a binary to a DGA automatically.

Rik Farrow asked how they assessed the numbers of infected machines in their case study. Manos replied that they considered the bot IPs, but looking at the bot host IDs they found that there was almost 1-1 matching. Another member of the audience asked how they got their data. Manos replied that ISPs come to Damballa to look for help in securing their

networks. Niels Provos asked whether it might be that those domains are not representative of botnets, but they are generated by malicious JavaScript. Manos said this is a possibility, but such clusters wouldn't be very big.

PUBCRAWL: Protecting Users and Businesses from CRAWLers

Gregoire Jacob, University of California, Santa Barbara/Telecom SudParis; Engin Kirda, Northeastern University; Christopher Kruegel and Giovanni Vigna, University of California, Santa Barbara

Gregoire Jacob presented their work on detecting Web crawlers. Web crawlers are a big problem, because they both generate overhead on the Web site, and collect personal information, leading to information leaks. Current solutions include using a robots.txt file, requiring logins on the sites, or displaying CAPTCHAs. However, these techniques are not always effective or applicable.

The authors proposed a system to detect crawlers, called PubCrawl. PubCrawl leverages a simple observation: regular users show daily regularity in the access to a site, and high versatility in the long term, while crawlers show high stability and versatility in the long term. For this reason, the system looks at time series to detect crawlers.

Greg explained how the system first applies heuristics to detect suspicious HTTP header fields, or anomalies in the sequence profiles of accesses. Then, they apply more complex techniques, involving autocorrelation and decomposition of the time series. The system detects similarities among the time series and performs clustering, to group together clients that had similar access patterns.

A limitation of the system is that it cannot make a decision if the time series is short. To mitigate this, the authors display a CAPTCHA on the Web site to slow down a possible crawler.

For the evaluation, they leveraged a data set from a large social network. The naïve heuristics exposed could detect only 47% of the crawlers, while using the time-series techniques allowed them to detect 98%. The cases in which the system would have been unsure were very few, and only 0.1% of users would have ever have received a CAPTCHA to solve. The system is able to detect distributed crawlers, too. PubCrawl might not be able to detect all of them as a single crawler, but they would still be detected as separate ones. The system has now been deployed on a social network system, and is used in production.

Shane Clark (U Mass) asked whether the detection changes significantly based on the type of the crawler. Greg replied that the detection is influenced by pure statistics. Stephen Huntler (UCI) asked whether he had any idea how many legitimate crawlers were in the data set. Greg replied that he didn't know, but that such crawlers could be whitelisted depending on the Web site policy. Yazan Boshmaf (UBC) asked about false positives. Greg replied that the system had an accuracy of 82%. The high number of false positives was because the system did not observe enough data to make a decision in many cases.

Enemy of the State: A State-Aware Black-Box Web Vulnerability Scanner

Adam Doupé, Ludovico Cavedon, Christopher Kruegel, and Giovanni Vigna, University of California, Santa Barbara

Adam Doupé presented their work on inferring Web application state and using it for vulnerability analysis. The reason for this is simple. Both white-box and black-box scanners have problems when finding vulnerabilities on Web applications. In particular, these systems ignore the state of the application, and this results in a very low code coverage.

To overcome this problem, Adam and his colleagues developed a system to infer the state of a Web application. To do this, they modeled a Web application as a symbolic Mealy machine. They then fuzz the application to infer its state machine. They detect a state change when the same input generates two different outputs at two points in time. When they detect a state change, they apply a heuristic to detect which request is actually responsible for the state change. After having built the state machine, the state machine is not minimal. Therefore, they need to collapse together states that are actually the same. This problem is reduced to a graph coloring problem.

After having inferred a Web application state machine, they leverage this knowledge for fuzzing. To this end, they try to fuzz every possible request in each state. Whenever they observe a state change, they reset the Web application and start over.

They evaluated their tool against skipfish, w3af, and wget. For their tool, they used w3af's fuzzer to aid their state-aware fuzzing. Therefore, any improvement on w3af is due to their state-aware analysis, and not of the fuzzer. They analyzed eight different Web applications, of various complexity, between 800 and 110,000 lines of code. They showed how their code coverage is better than any other tool, covering up to 150% more code. Due to this improvement, they were able to find three vulnerabilities in the studied applications that no other tool was able to find.

A member of the audience asked whether they tried to compare their tool with commercial ones. Adam said they did not, because commercial vulnerability scanner companies don't like them. Ben Livshits (MSR) pointed out that this approach doesn't work for single-page JavaScript applications. Adam replied that one could convert AJAX apps to something more static or rethink what you consider as a request. Someone

asked what would happen if they applied their technique to white-box approaches. Adam answered that this would make the approach more effective.

Invited Talk

Summarized by Michael Dietz (mdietz@rice.edu)

Emerging Trends and Their Security Implications

Doug Burger, Director, Client and Cloud Applications, Microsoft

It's hard to argue against the acceleration of change occurring in the computing world. In just five short years since 2007 we've gone from early generation smartphones to the cheap, powerful, and ubiquitous mobile devices that we have today. Doug Burger, director of client and cloud apps at Microsoft, goes on to envision the next few years when the age of mobile computing will give way to the era of "continuous computing." Burger first praised the advances that we've made in the mobile computing space but vilified the user experience that mobile devices provide. He claimed that they merely provide a tiny viewscreen into the digital world and have very little sense of the semantic meaning of the world around them. A truly "continuous" computing experience would change all that and provide a frictionless, personalized experience that's tightly coupled with our human sense.

To expand upon his view of the world to come Burger outlined five layers of the continuous computing environment, from interface devices connected to your body to the "far cloud" that does the heavy lifting, aggregating data and managing the caching and deployment of services to the four layers below it. The most interesting aspect of this continuous computing model is the speculation that a loosely coupled collection of personal-area computers backed by a smart cloud could make decisions based on a rich understanding of the user's environment and requirements (e.g., not delivering a phone call to Burger during his talk unless it's his wife).

Burger then jumped into the security ramifications of his continuous computing model. One of the most pressing questions is, who owns the data gathered by sensors? Sensors that are installed in a private residence and the data they produce presumably belong to the owner, but what about sensors in an office building that report to an employee's personal device? Does the employee's company own that data? These ownership overlaps present a huge risk for privacy violations in a continuously connected world.

Burger then proposed the beginnings of a solution to the security risks presented by continuous computing: a tight coupling of data with policy in order to control the "digital exhaust" that we create even today (via tracking cookies, targeted ads, etc.). In order to achieve this tight coupling we need both strong policy, such as that provided by the SecPal project from Microsoft Research, and enforcement mechanisms of those policies on the data consumer side of the equation. Burger also suggested that we need to move cloud services from the "far" cloud, where they are out of our control, to a "near" cloud that's local to a single person and allows for better privacy management.

While the first half of Burger's talk covered his vision for the future, the second half dealt with the practicalities of how we'll get there. Burger discussed the imminent disruption in silicon brought on by the end of Moore's Law. Additionally, the constant power usage that we've seen as transistor density increased is also at an end. This means that either some new technology will need to be discovered to maintain the current rate of computing performance growth in the next decade, or our approach to how computing systems are built will need to drastically change.

Burger presented several possible directions for a new approach to computation. The first—producing lots of cheap, low-power chips and installing them in everything—meshes with the notion of continuous computing. The other possible approaches are more in line with traditional thinking, from system-on-a-chips, which integrate motherboard and CPU into a single chip, to extreme specialization, where a custom chip is fabricated on demand for a particular task and installed on a machine responsible for just that task. Burger concluded by pointing out that all of these approaches have their own set of security risks, from questions of what happens when every light bulb can execute arbitrary code to whether you can trust the manufacturer who fabricates a specialized chip to refrain from installing hardware back doors.

Guerney Hunt (IBM) asked Burger to consider the disruptive materials and technologies that he had dismissed earlier in the talk. Burger responded that the Graphene work that IBM is doing is interesting. He then classified these disruptive technologies as silicon-like and "other," where technologies in the "other" category, such as quantum computing, are ten plus years out and unlikely to affect the current silicon roadmap.

Mobile Devices

Summarized by Aaron Blankstein (ablankst@cs.princeton.edu)

Aurasium: Practical Policy Enforcement for Android Applications

Rubin Xu, Computer Laboratory, University of Cambridge; Hassen Saïdi, Computer Science Laboratory, SRI International; Ross Anderson, Computer Laboratory, University of Cambridge

Rubin Xu explained that practical policy enforcement on the Android platform is important in preventing malicious applications from compromising user privacy, and that existing protections on devices are not sufficient with previous research requiring extensive modifications to the OS. He

then presented Aurasium, a system for providing policy enforcement on Android that does not require modifying the OS while still providing much of the security and privacy that users desire.

Aurasium automatically repackages arbitrary applications to attach user-level sandboxing and policy enforcement code, which closely watches the application's behavior for security and privacy violations. Aurasium can also detect and prevent cases of privilege escalation attacks. Experiments showed that Aurasium works on a large sample of benign and malicious applications with a near 100% success rate and without significant performance and space overhead. Aurasium was tested on three versions of the Android OS and is freely available.

Jaeyeon Jung (Microsoft Research) asked about whether information flow policies could be expressed with this system. Xu responded that you could enforce coarse-grained information policies. For example, you could mark a file as private, but that would not be a fine-grained information flow policy. Someone asked how users would interact with this system and create policies. Xu replied that users currently respond to pop-ups that request permissions, but that policies could also figure these things out automatically. Policies would be written by experts, not the users. Finally, someone asked whether this repackaging system requires enabling an untrusted source and did this pose a security risk. Xu said that, yes, this is somewhat of a problem, but there is plenty of malware on the app store that does not require enabling untrusted sources.

AdSplit: Separating Smartphone Advertising from Applications

Shashi Shekhar, Michael Dietz, and Dan S. Wallach, Rice University

Dan Wallach discussed how smartphone applications rely on third-party advertising services, which provide libraries that are linked into the hosting application. This situation is undesirable for both the application author and the advertiser. For the application author, advertising libraries require their own permissions, resulting in additional permission requests to users, what Wallach called permission bloat. The researchers found 26% of applications could require at least one fewer permission to run without advertisements. The existing system is bad for the advertisers because malicious apps can simulate the behavior of the advertising library, forging the user's interaction and stealing money from the advertiser.

The researchers presented AdSplit, a system to extend Android to allow an application and its advertising to run as separate processes, under separate user IDs, eliminating the need for applications to request permissions on behalf of their advertising libraries, and providing services to validate the legitimacy of clicks, locally and remotely. AdSplit automatically recompiles apps to extract their ad services with minimal runtime overhead. AdSplit also supports a system resource that allows advertisements to display their content in an embedded HTML widget, without requiring any native code.

Wallach summarized future possibilities as native mobile code and mobile Web applications converge. The session chair, Will Enck from North Carolina State University, asked about analytics libraries, which are often used in addition to ad libraries. Dan responded that he had not thought about analytics but that his gut reaction is that analytic libraries come in many different forms. Some have code in many different places, all over the application, while some are simply small additions. These would have different implications for how they get handled. Ben Livshits (Microsoft Research) commented that users do not actually understand permissions. Wallach replied that if you look at the markets, apps that require lots of privileges will see lots of user feedback about those privileges. Livshits asked what incentives are provided to the various players. Wallach responded that AdSplit is beneficial to all players—it helps prevent click fraud and it lowers the privilege required for applications.

DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis

Lok Kwong Yan, Syracuse University and Air Force Research Laboratory; Heng Yin, Syracuse University

Lok Kwong Yan began by pointing out that the prevalence of mobile platforms, the large market share of Android, and the openness of the Android Market make it a hot target for malware attacks. Once a malware sample has been identified, he explained, it is critical to quickly reveal its malicious intent and inner workings.

The researchers presented DroidScope, an Android analysis platform that uses virtualization-based malware analysis. Unlike desktop malware analysis platforms, DroidScope reconstructs both the OS-level and Java-level semantics. To facilitate custom analysis, DroidScope exports three tiered APIs that mirror the three levels of an Android device: hardware, OS, and Dalvik Virtual Machine. The researchers developed several analysis tools on top of DroidScope to collect detailed native and Dalvik instruction traces, profile API-level activity, and track information leakage through both the Java and native components using taint analysis. These tools proved to be effective in analyzing real-world malware samples and incur reasonably low performance overheads.

Jaeyeon Jung (Microsoft Research) asked whether this instruction-level analysis allowed the researchers to find anything that TaintDroid (another Android analysis tool) missed. Yan responded that they had not found anything missed by TaintDroid, but that they had not looked at a large corpus of applications. Will Enck (North Carolina State University) asked whether OS changes will make their system rather fragile. Yan responded that they had not seen many changes that affect the code greatly—the shadow memory map that they use allows for some abstraction between changes to the implementation and their system. Ben Livshits (Microsoft Research) asked if any support from the Dalvik team would help. The presenter responded that better support for tracing through JIT would help greatly. Yan was also asked if cryptographic libraries would disrupt the way that data is monitored, to which he responded that taint analysis runs through cryptographic libraries very well, particularly because these libraries do not over-taint.

Software Security II

Summarized by Shirin Nilizadeh (shirnili@indiana.edu)

STING: Finding Name Resolution Vulnerabilities in Programs

Hayawardh Vijayakumar, Joshua Schiffman, and Trent Jaeger, The Pennsylvania State University

Hayawardh Vijayakumar started by explaining file system DNS and two types of attacks: (1) improper binding attack, when an attacker introduces bindings to resources outside of the attacker's control, and (2) improper resource attack, when an attacker creates an unexpected binding to a resource that the adversary controls. Both give an ability to the low integrity adversary process to share the same OS namespaces as high integrity victim processes. Although these attacks are local exploits, the adversary can be either an untrusted local user in a multi-user environment (e.g., university) or a remote attacker who has intruded onto the network. These attacks also leverage the non-atomicity of various system calls to create races, such as the time-of-check-to-time-of-use (TOCTTOU) attacks. Vijayakumar explained that this problem is serious since these attacks contribute 5–10% of CVE entries each year, and finding these vulnerabilities is difficult. One interesting argument was that although it is expected that the programmers find these vulnerabilities, it is very difficult for them because the vectors for name resolution attacks are outside the program.

The existing defenses are based on dynamic and static analysis. These are not effective and have a high false positive rate. The authors have developed STING, which actively detects these vulnerabilities. STING has two phases: a launching phase and a detecting phase. The launching phase is invoked at the start of a name resolution system call. At this phase, at first, the bindings of the name are normally resolved, and then it is checked to determine whether any attack is possible. If an attack is possible, STING chooses an attack from the pre-defined list of attacks, generates a test case resource, and launches the attack. The detect phase is invoked on accept system calls when a victim process has accepted STING's changes. Detecting the vulnerability, STING records its information and reverts the namespace to avoid side effects.

STING is implemented in the Linux 3.2.0 kernel. It was tested on Ubuntu 11.10 and Fedora 16. Under the DAC attacker model, STING found 26 vulnerable resolutions across 20 distinct programs. STING was able to find vulnerabilities of all types, including seven that required race conditions. Under the MAC adversary model, STING reported vulnerable name resolutions whenever any program accessed /etc because the SELinux policy allows untrusted subjects to create permissions to critical labels such as etc_t.

At the end, it was concluded that it is both difficult to prevent name resolution attacks and find program vulnerabilities, so there is a need for greater cooperation between programmers and distributors. Vijayakumar also mentioned that the resources are available online. Deskin Miller (Microsoft) asked if it is really required for low integrity and high integrity processes to interact with each other. Vijayakumar answered that for functionally purposes it is necessary.

Tracking Rootkit Footprints with a Practical Memory Analysis System

Weidong Cui and Marcus Peinado, Microsoft Research; Zhilei Xu, Massachusetts Institute of Technology; Ellick Chan, University of Illinois at Urbana-Champaign

Weidong Cui explained the kernel rootkit security threat that can compromise the OS kernel and own the entire software stack. To detect and analyze a kernel rootkit, one needs to identify a rootkit's memory footprint, and to do that one needs to check the integrity of both static and dynamic data. However, checking the integrity of dynamic data is a challenge. The existing solutions use the basic memory traversal procedure where in order to locate dynamic data, first static data objects are located, then recursively the pointers in these objects are followed until no new data object can be added. However, this procedure has a practical problem that has not been addressed: if a pointer is invalid, the error will be propagated and accumulated. Pointer uncertainty is unavoidable because of invalid pointers and ambiguous pointers. It was argued that to decrease the false positives and false negatives of these solutions, one needs to handle pointer uncertainty. Pointer uncertainty can be handled by (1) identifying data objects without following pointers, (2) ignoring pointers with ambiguous types, (3) checking all vulnerable constraints before following them, and (4) employing

some type constraints. Each of these was explained in detail during the presentation.

In summary, MAS identifies all memory changes that a rootkit makes in three steps: static analysis, memory traversal, and integrity checking. For static analysis, MAS uses a pointer analysis algorithm to identify candidate types for generic pointers. In memory traversal, MAS identifies the dynamic data objects in a given memory snapshot. For integrity checking, MAS identifies the memory changes that a rootkit makes by inspecting the integrity of code, static data, and dynamic data. Finally, it outputs the list of identified integrity violations.

Cui mentioned that although the design seems simple, implementation was very hard with 12,000 lines of C++ code. MAS was evaluated through its accuracy, robustness, and performance. The evaluation was performed on three sets of real-world memory snapshots and crash dumps. The experiments show that MAS was able to quickly analyze all memory snapshots with running times between 30 and 160 seconds per snapshot and with near perfect accuracy. It could also quickly identify 95 crash dumps that contained rootkits.

Someone from Columbia University asked whether it is a general fact that more than 10% of crashes are caused by rootkits or whether this percentage depends on data sampling. Cui responded that they have not sampled the memory snapshots but used all the memory snapshots that were collected in three months.

Tachyon: Tandem Execution for Efficient Live Patch Testing

Matthew Maurer and David Brumley, Carnegie Mellon University

Maurer noted that many attacks have occurred when the vulnerability has been known and a patch was available but not installed. He argued that patch installation is often delayed because patches must be tested manually. To minimize this delay, the authors have proposed using a new tandem execution approach that tests patches automatically. Their solution is based on the patch execution consistency model that a patch is safe to apply if the executions of the pre- and post-patch program only differ on attack inputs. The idea is that the patched program runs live on the system with all syscalls being serviced by the kernel and, simultaneously, the unpatched program runs in tandem, but with each syscall to the kernel simulated by replaying the side effects from the corresponding calls of the live version.

Maurer explained that they implemented their approach in a system called TACHYON that is based on syscall replay techniques for binary programs. TACHYON works with semantics rather than capturing details such as pointer values that may differ between patches. By observing semantic deviations between executions of two programs, they will be recorded and the user will be informed about them. These deviations can be either the changes in the actual inputs and outputs or the changes in the sequence of syscalls. To identify the first kind of deviation, they extended C side effects, which enables TACHYON to identify semantically meaningful arguments. To identify the later deviations, a lightweight rule-based system has been proposed that checks the syscall stream equivalence when the sequence of syscalls between the patched and unpatched binary are not exactly the same. To continue testing, the user needs to provide a rewrite rule that indicates how the deviation can be handled. In the experiments, it was shown that rewrite rules are small when needed, and often completely unnecessary for security-related fixes.

To show the effectiveness and performance of their system, they evaluated their approach on several real-world patches and synthetic benchmarks. They tested TACHYON on the most recent 207 patches to coreutils. It was found that in 18 cases out of 1656 executions, a deviation was reported, or TACHYON crashed, where 16 of these were TACHYON bugs and two were actual deviations. The overhead of TACHYON was also evaluated and it was shown that both data transfer and system calls had a linear overhead and the CPU overhead was 0%.

Maurer also emphasized that TACHYON can be used for other purposes such as honeypots, and debugging, which he explained in detail. He also mentioned some of the limitations of TACHYON. For example, it only runs on Linux and it does not support virtual dynamically linked shared objects. Someone argued that most of the delay in installing patches is because users do not want to restart their laptops. Maurer responded that TACHYON does not target home users but targeted users at large organizations.

Being Social

Summarized by Alexandros Kapravelos (kapravel@cs.ucsb.edu)

Privacy-Preserving Social Plugins

Georgios Kontaxis, Michalis Polychronakis, and Angelos D. Keromytis, Columbia University; Evangelos P. Markatos, FORTH-ICS

Georgios Kontaxis dealt with the dilemma of social plugins vs. user privacy. Social plugins are embraced by more than 35% of the top 10,000 most visited Web sites, meaning that the social platform provider has enough information to associate users to Web site visits. Current approaches to social plugins' privacy issues are limited, since they eliminate user tracking but in the process sacrifice the user experience.

The key contribution is that Georgios decouples the retrieval of user-specific content from the loading of the social plugin. This way personalized content is still available to the user without revealing to the social platform provider any user-

specific information. This is achieved by maintaining a local copy of all private user information, such as names and pictures of friends, and requesting anonymously from the social network platform only public information, such as the total number of "likes."

The authors implemented their approach as an add-on for Firefox called SafeButton and made it publicly available. This enables users to take action and protect their privacy if they want to, without sacrificing the social aspect of the Web. Interestingly enough, the plugin is faster than loading the original Like button because the former is using the Graph API to query for the total number of "likes" and the latter is fetching a full HTML page with embedded CSS and JavaScript content. Another great aspect of their approach is that it can be implemented as a service, eliminating the users' need to take any action to protect their privacy.

Social Networking with Frientegrity: Privacy and Integrity with an Untrusted Provider

Ariel J. Feldman, Aaron Blankstein, Michael J. Freedman, and Edward W. Felten, Princeton University

▶ ***Awarded Best Student Paper!***

Ariel Feldman presented work about untrusted social network providers. Today we entrust to the providers of social networks that we use very sensitive information. In addition we take the integrity of our messages for granted. Prior work that copes with untrusted providers in social networks has either proposed a decentralized approach or did not take into consideration integrity. Frientegrity is a novel social network framework that guarantees both privacy and integrity without sacrificing performance.

Frientegrity relies on users verifying the content served by the provider. By design the user can verify efficiently that the provider did not tamper with the content, the content served was created by an authorized user, the provider has not equivocated about the set of authorized users, and the ACL (access control list) is not outdated. In fact users collaborate to verify object histories and ensure fork* consistency based on history trees. A well-defined API is provided to clients to interact with the system which permits them to read and write content and manipulate ACLs.

To evaluate Frientegrity the authors implemented a prototype that mimics a simplified Facebook-like service. The system managed to achieve a read/write latency of approximately 6 ms/10 ms, respectively on single objects. The network overhead of the additional data that needed to be sent, such as signatures, was found to be only 11,300 bytes. In total, Frientegrity managed to achieve response times that are satisfactory for interactive use without sacrificing security and integrity of the system.

Efficient and Scalable Socware Detection in Online Social Networks

Md Sazzadur Rahman, Ting-Kai Huang, Harsha V. Madhyastha, and Michalis Faloutsos, University of California, Riverside

Md Sazzadur Rahman presented his work on a new upcoming threat called socware, a term that they use to refer to spam and malware on social platforms. Socware differs from traditional email spam and malware: it propagates through its victims ("liking" or sharing posts), and it is sometimes hosted on the social network provider itself, rendering URL blacklists and DNS reputation techniques ineffective.

The authors have implemented a Facebook application called MyPageKeeper, which detects socware and protects Facebook users. The application works by collecting wall posts with URLs from its users, extracting some features which take into consideration the social context of the post (i.e., the number of "likes"). Based on this input from multiple users, MyPageKeeper classifies the URL as socware according to a machine-learning classifier. If a socware post was detected, the user is notified and in the future the application will also remove malicious posts.

To evaluate MyPageKeeper the authors ran the application for four months with 12,000 users. Over this time they examined 753,516 URLs and classified 4,972 of them as socware. They used additional features to build ground truth, such as deleted domains and blacklisted IPs, and found 97% of the socware to be true positives. Since MyPageKeeper is using only the social context to determine if a URL is socware or not, it was also more efficient. In fact the throughput is an order of magnitude greater than resolving the corresponding complete URL from a short URL and querying a local blacklist.

Invited Talk

Summarized by Aaron Alva (aalva@uw.edu aalva@uw.edu)

Securing Early Software Development

Riley Eller (Caezar), Security Strategist for Leviathan Security Group

Riley aptly summarized his talk with a penultimate slide: retain a security advisor with a fixed quarterly engagement; use the advisor to plan security development by following a maturity model; keep to a fixed budget and expect variable timelines. This approach will slowly nurture security practices from within.

Riley began his talk with a brief background of his 20-plus years in security development and how security has not happened in all of them. There is a persistent set of problems in how software development is funded that has limited the application of research, such as research presented at USENIX, causing it to not be implemented. —Riley demonstrated his breadth of experience (in small businesses, in the hacker community, and in developing the first generic fuzzer)

while also confessing that he has never worked in a secure environment.

The problem with telling people in small team settings to make any fix is that they aren't going to do it; things come up such as consumer demands and business plans. NIST says that security comes first and you can't build in security after you build in a project, while others say you can't hire security (Earl Boebart, ex-NSA) and you can't build security from within unless it is a core focus (InfoWorld). But the reality is that security in small teams comes last. Riley's talk applies not just to small businesses but also to small teams within larger corporations.

He was asked by a friend to consult on how to make a payment card system better, and for the first time he considered the ethics of having whatever he wrote put in a folder. The question became: "Why can't I ethically consider myself as an advisor when they give me money?" Realistically, small teams don't have time and cannot afford to think about security; it costs more to tell the manager to deliver something late than a compromise at a released point of sale terminal. In addition to the time constraints, there are attention issues to consider.

So how do you get the security needed despite all of the barriers? It's clear that the goal should be to develop security practices so that by the time your small team becomes a target, you are strong enough to survive. The solution is to externalize security. Hire someone much like Riley who will come in once a quarter and have a conversation. Make this part of your budget. Treat the security advisor much like you would treat outside legal counsel. Build this relationship, and slowly cultivate security skills from within.

To manage costs, keep a fixed budget. Consider 0.5% of revenue, and negotiate a fixed price for quarterly steering meetings with the advisor. Attempt to implement each Statement of Work recommended by the advisor in-house before collecting bids; down the road it may be possible to fire the security advisor and hire from within because you've been increasing security.

And how can the security advisor plot security progress throughout the process? Riley has employed the "Building Security in Maturity Model" by Cigital (http://www.bsimm.com/) and its incremental levels. To get the ball rolling, begin the first few activities such as mapping network assets and interviewing staff to build an initial threat model, but stop before considering any technology. When you come back in a quarter, take on the next few steps piece by piece.

The result is a slow-cooked security program with a "virtual information security officer" as advisor. Riley's experience doing this with four companies thus far is radically different from the security development norm. The team feels like they are growing into shoes instead of being beat up by a consultant. The benefits include the NIST-mandated compliance; up to a 30x lifetime reduction in security bug costs if you build a security program from the beginning of the SDLC; and a growing staff competence from day one. To opt out of security development is unacceptable; opting in is impractical. This model proves the viability of a different approach to security development.

USENIX Security Symposium Chair Yoshi Kohno asked about the age of the model and whether other organizations were doing this. Riley said he wrote this model in September 2011 and is not aware that anyone else is using a retainer service even if they don't get services; many think of buying firewalls and doing other things instead. Yoshi than asked if this is a replacement for small teams or should they merge together? Riley's plan is to use the BSIMM as a reasonable milestone. Any implementation by a small team should be paired next to a maturity model.

Deskin Miller (Microsoft) asked what Riley would say to companies already in trouble. Riley cited Blizzard and Microsoft as examples—both have the money to buy out problems. Even with Trustworthy Computing, which started with Bill Gates' memo, it took all this time for Microsoft to get to their current security posture. But there's a large cost associated with not doing this in the 80s. As for companies that flash big, Riley hopes they've thought about security already. Zynga, for instance, needs to take a more aggressive strategy by rolling whole teams off main development and into security development, as covered in the BSIMM. The goal for the flash-big companies is to spend their way up to level two. Once the infrastructure is in place, they can work on security growth through the BSIMM approach.