Conference Reports

23rd USENIX Security Symposium
August 20–22, 2014, San Diego

Summarized by David Adrian, Qi Alfred Chen, Andrei Costin, Lucas Davi, Kevin P. Dyer, Rik Farrow, Grant Ho, Shouling Ji, Alexandros Kapravelos, Zhigong Li, Brendan Saltaformaggio, Ben Stock, Janos Szurdi, Johanna Ullrich, Venkatanathan Varadarajan, and Michael Zohner

Opening Remarks
Summarized by Rik Farrow (rik@usenix.org)

Kevin Fu, chair of the symposium, started off with the numbers: a 26% increase in the number of submissions for a total of 350, and 67 papers accepted, a rate of 19%. The large PC was perhaps not large enough, even with over 70 members, and included outside reviewers as well. There were 1340 paper reviews, with 1627 follow-up comments, and Fu had over 8,269 emails involving the symposium. Attendance was strong, with 520 people registered.

Jaeyeon Jung (Microsoft Research) worked as Fu's deputy chair, and will be the chair for the symposium in 2015. Tadoyoshi Kohno, past chair, handled both WiPs and the shadow reviewers.

Fu announced the Best Paper award, which went to "Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing," by Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Two Best Student Paper awards went to "DSCRETE: Automatic Rendering of Forensic Information from Memory Images via Application Logic Reuse," by Brendan Saltaformaggio, Zhongshu Gu, Xiangyu Zhang, and Dongyan Xu, and to "Automatically Detecting Vulnerable Websites Before They Turn Malicious," by Kyle Soska and Nicolas Christin.

The Test of Time award, for a paper that was presented at least 10 years ago and is very relevant today, went to Roger Dingledine, Nick Mathewson, and Paul Syverson for "Tor: The Second-Generation Onion Router." Dingledine and Syverson were present and received the award, and also spoke briefly in a short panel after the keynote.

Keynote
Summarized by Rik Farrow (rik@usenix.org)

Phone Phreaks: What We Can Learn from the First Network Hackers
Phil Lapsley, hacker, consultant, entrepreneur, and author of *Exploding the Phone: The Untold Story of the Teenagers and Outlaws Who Hacked Ma Bell.*

Phil Lapsley told us that he spent five years researching and writing his book, including over 500 FOIA requests. He later suggested looking at the Web site for the book, where you can search through his references (explodingthephone.com), something I tried while writing this summary, and I can write that searching worked well for me. And while the book is about phreaks, Bell System hackers, Phil presented the background that made phone freaking not just possible but likely.

AT&T was a US government regulated monopoly, with over 90% of the telephone market in the US. They owned everything, including the phones on desktops and in homes, and, as a monopoly, could and did charge as much as $5 per minute (in 1950) for a cross-country call. In 2014 dollars, that's nearly $50 per minute. In the beginning, the system relied on women, sitting in front of switchboards with patch cables in hand, to connect calls. Just like Google today, AT&T realized that this model, relying on human labor, cannot scale with the growth in the use of the telephone: They would have needed one million operators by 1970. Faced with this issue, company researchers and engineers developed electro-mechanical switches to replace operators.

The crossbar switch, first used only for local calls, used the pulse of the old rotary telephone to manipulate a Strowger switch, with each pulse setting the switch to the next position. For long-distance calls, operators would have to find a set of available trunk lines to route each call. AT&T's next step was to build more automated switches, called a 4A crossbar, each so large it filled a city block, eventually building over 175 of them. As these 4A crossbars were networked together, they together comprised the largest machine ever built.

The 4A crossbar used a tone, 2600 hertz, to indicate when a trunk line was idle, and pulses to communicate across the trunk to a distant 4A crossbar. The first "hack" of the system occurred when a blind 13-year-old, Joe Engressia, was whistling along with a song and noticed that it caused a phone call to be disconnected. He experimented by calling information (dialing 411), then whistling again (he had perfect pitch), disconnecting that call as well. With practice, he could "dial" long distance calls just by whistling, and Lapsley displayed an old TV segment of an older Engressia doing just that. And, said Engressia on TV, the phone call was free, too. He had discovered the interface used by the switches, and by operators.

AT&T had a big problem. Fixing the system would be very expensive, so they tried to interest the FBI in prosecuting people using the 2600 Hz tone to make free calls, but the FBI wasn't interested at first. Bookmakers, who arranged bets, made a lot of use of phones, and the ability to make both toll free and unlogged phone calls really appealed to them. By collecting information about bookies using the 2600 Hz tone, the phone company got the FBI's attention.

Generating the 2600 Hz tone, along with the other seven tones used with switching equipment, took a little bit of technical know-how, provided by telephone company documents. The first devices were large, and built into blue boxes, which is where the devices got their name. Steve Jobs and Steve Wozniak learned about blue boxes, and sold them to students living in UC Berkeley dorms. This was their first entrepreneurial enterprise, in 1971, after learning about the hack via an article in *Esquire* magazine. The phone company countered by building green boxes, devices for detecting the presence of 2600 Hz tones entering switches from locations other than trunk lines or 4A crossbars.

Lapsley ended with a list of observations, such as, if you build it, curious kids will hack it, and later so will organized crime and state actors. Bell Labs can be forgiven for not considering security, because there weren't hackers back then. But we now know better. It's better to build security in from the start, rather than attempting to bolt it on later. AT&T had the 175 gigantic 4A crossbars, plus many thousands of smaller switches, all relying on the 2600 Hz signal. In closing, Lapsley said that if you think things are bad now, just wait: The Internet of Things means that there will soon be trillions of connected devices, all with little to no security designed in.

Vern Paxson praised the talk, then pointed out the mission creep: A search for phreakers turns into an effort to help locate bookies. Lapsley said he had interviewed Bill Caming, a fraud attorney for AT&T, who said the FBI was quite happy to receive a comprehensive list of bookies, and that it was part of an ongoing relationship between AT&T and the government. Paxson went on to say that mission creep was a subset of co-evolution, and Lapsley responded that AT&T was a Stalinish Central Planning organization, but one that doesn't exist any more. Steve Bellovin, speaking as a historical researcher, really appreciated the online references (see above). Bellovin then pointed out that the FBI wasn't interested in cloned cell phones either, until they learned that drug dealers were using them. John Stalwart said he was interested in the special numbers, 000–199, that Lapsley had mentioned. These numbers allowed phone company insiders to test lines, but also to eavesdrop on existing connections, and that because there were so many people employed, it was possible to find someone to bribe.

Tor Panel and Lightning Talks

The summary of this session is available online as an electronic supplement: www.usenix.org/login/dec14.

Privacy Session

Summarized by Kevin P. Dyer (kpdyer@gmail.com)

Privee: An Architecture for Automatically Analyzing Web Privacy Policies

Sebastian Zimmeck and Steven M. Bellovin, Columbia University

Sebastian presented a very relatable problem: Privacy policies are often hard to understand and long to read. Most users simply browse a Web site or click "accept" without appreciating the implications of their actions. What's more, in many jurisdictions, such as the United States, privacy policies are legally binding documents. In response, the authors present Privee, a concept that helps users understand the privacy policies they agree to.

Privee is implemented as a browser extension and has two methods for analyzing policies. First, the browser extension checks whether a crowd-sourced analysis of a given privacy policy exists in a public repository. If a crowd-sourced analysis of the policy is not available, rule and machine-learning classifiers are used to dynamically analyze and classify the policy. At the end of this work-flow, the results are displayed in a simple UI overlaying the complex policy. The UI reports on six different binary dimensions that are of interest to users, such as: "Does the company encrypt my data when in transit and/or storage?" or "Does the policy allow for collection of personal information?"

A ground truth data set to evaluate the accuracy of Privee was obtained by having experts tag privacy policies. It turns out that the accuracy of the machine-learning-based classification is dependent upon the type of question being asked. As an example, it's typically quite easy to infer the context and meaning of words like "encryption." However, other words like "disclosure" turn out to be problematic for binary classifiers. Interestingly, both human experts and the Privee extension had more difficulties with ambiguous policy language. Nevertheless, Sebastian was optimistic that there will be future improvement in the classification process. He emphasized that privacy policies tend to become easier to understand and classify over time. This all points to Privee being a very promising approach to empower users.

During the Q&A, someone asked how Privee and its classifiers compared to P3P classifications of privacy policies. Sebastian responded that the P3P privacy policies of Web sites are, in fact, often wrong and, thus, diverge from natural-language policies. However, some of the classifications performed by the Privee extension are comparable to the P3P tags, which hints that the Privee concept might have the same expressive power as P3P.

Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing

Matthew Fredrikson, Eric Lantz, and Somesh Jha, University of Wisconsin—Madison; Simon Lin, Marshfield Clinic Research Foundation; David Page and Thomas Ristenpart, University of Wisconsin—Madison

Awarded Best Paper!

Matthew started by describing an area of medicine that we may not all be familiar with: pharmacogenetics. Pharmacogenetics is the use of patients' health records and genetic information to provide individually tailored drug dosing. For certain drugs like Warfarin, a blood thinner, this is critical—incorrect dosing can kill a patient. For this reason the International Warfarin Pharmacogenetics Consortium (IWPC) publishes a pharmacogenetics-based model that works well for tailoring initial Warfarin dosing to a patient, based on the patient's age, height, weight, race, history, and two specific genotypes.

In the medical community, data sets are rarely published. However, the models derived from data sets are often published. In response, the authors present a novel attack, which they call a model inversion attack. In the case of the IWPC model, the model inversion attack means that given just the model, it's possible to predict the genotype of a specific patient. It is assumed that the attacker has basic demographics, stable Warfarin dose, black-box access to the model, and marginal priors on patient distribution. Then it turns out that an attacker can use model inversion to infer patients' genotypes based on this data more accurately than guessing based on the priors. The accuracy is nearly optimal, and they provide a detailed argument in the paper.

A natural approach to combating a model inversion attack is to apply differential privacy. Ideally, this would allow us to maximize the accuracy of queries but minimize the chance any specific patients' information could be recovered from the model. However, differential privacy is implemented by adding noise to the underlying data set. As it turns out, this changes the model, which, in turn, changes the initial dosing that a patient would receive. Matthew highlighted that there is now an undesirable tension: When differential privacy is applied, patients are at increased risk of negative outcomes, including mortality, and when differential privacy is not applied, patients' privacy is at risk.

The question and answer session started with a question about diet: There is a complex interaction between Warfarin dosing and a patient's diet. Matthew responded that diet was beyond the scope of this study and not considered in the model. Someone asked what the general feeling in the medical field was with respect to privacy. Matthew, of course, couldn't speak for the medical field in general but said that the physician on this project thought that adding noise to the underlying data set was a bizarre strategy for achieving privacy.

Mimesis Aegis: A Mimicry Privacy Shield—A System's Approach to Data Privacy on Public Cloud

Billy Lau, Simon Chung, Chengyu Song, Yeongjin Jang, Wenke Lee, and Alexandra Boldyreva, Georgia Institute of Technology

Billy started by highlighting the tension between usability and security that many users grapple with. Users want the convenience of applications such as Gmail, Facebook Chat, or WhatsApp but don't necessarily want their private data to be stored in plaintext in the cloud. Mimesis Aegis (M-Aegis) addresses this concern by providing developers a framework for adding end-to-end encryption to applications. However, the key difference benefit of M-Aegis over other solutions is that it doesn't require modifications to or repacking of applications. M-Aegis takes advantage of the accessibility layer that is available on modern mobile operating systems. This is used to create a layer called Layer 7.5 (L-7.5) because it acts as a proxy between Layer 7 (application) and Layer 8 (user) of the OSI network model.

The key challenges in implementing M-Aegis is to ensure user experience is not compromised. Features such as spell check, in-app navigation, and search must be retained, despite the use of end-to-end encryption. As it turns out, using the L-7.5 approach, spell check and in-app navigation are features that were easy to retain on OSes like Android without additional infrastructure. However, search turned out to be slightly more problematic, because server-side cooperation cannot be assumed. To overcome this issue they adapted a searchable encryption scheme that was presented in a prior work; more details appear in the paper.

Using M-Aegis to implement end-to-end encryption requires per-app engineering to ensure that the proxy layer, L-7.5, correctly captures user input, encrypts it, then relays it to the underlying app. What's more, this additional layer plus encryption requires per-app decisions on how to encode data to ensure that the app correctly accepts the input, because not all apps can handle arbitrary ciphertexts.

Finally, to confirm that M-Aegis does not negatively impact user experience, a user study was performed. It was confirmed that no UI anomalies or performance issues were noticed by the users, despite the new layer of encryption.

Someone asked whether the framework required per-app manual work. Billy said that, yes, per-app engineering is needed, but the framework assists with this process. However, challenges may arise if an app doesn't use native UI rendering. Another questioner asked for clarification of the attack model: Is the attacker assumed to be honest but curious or active? Billy confirmed that an honest but curious adversary was assumed. Someone wondered whether most Android apps use the native UI. Billy confirmed that most Android apps do use the native UI in his experience. Were there any plans to support encrypted images? Billy responded that it's easy to encrypt text, but unless the ciphertext abides by specific formats, it is going to fail when stored. What's more, we need encryption schemes that survive compression. Finally, someone asked who manages keys. Billy replied that M-Aegis has a pluggable key distribution system, and any distribution strategy can be used.

XRay: Enhancing the Web's Transparency with Differential Correlation

Mathias Lécuyer, Guillaume Ducoffe, Francis Lan, Andrei Papancea, Theofilos Petsios, Riley Spahn, Augustin Chaintreau, and Roxana Geambasu, Columbia University

Mathias highlighted the issue of targeted advertising. Companies are aggressive, but not transparent, in how they target users. In some cases, companies may try to determine when you're sick, or they may even try to determine when you're pregnant. However, the ads displayed in response to these detected events don't always make it clear what's being targeted. So Mathias posed the question: Is it possible to construct a generic tool that reveals data misuse? As an example, can we determine which emails sent or received trigger a specific ad displayed to a user?

XRay is the first generic tool that correlates inputs (e.g., emails, Web queries, etc.) to ads output to a user. It turns out that the naive approach of creating many shadow accounts does not scale to address this problem. Therefore, the authors present two novel logarithmic algorithms for correlating user inputs with ads output. One algorithm is simple but not robust. The other algorithm is more complex, but more robust.

The algorithms were evaluated against Amazon, YouTube, and Gmail and do not assume that all ads are targeted. A logarithmic number of shadow accounts for each service were required to perform the evaluation. For Amazon and YouTube, when ads are displayed a reason is given (i.e., "We've shown you X because Y."). This enabled a ground truth data set to determine the classification accuracy. The authors' algorithms were then applied to identify more subtle advertisements on Gmail through manual labeling. This surfaced interesting results: for example, if someone is having financial problems, they are targeted with ads from subprime lenders.

The question and answer session started with an important question: What should users do with this information? Mathias responded that this is first time they had this information, so they didn't know yet. An ultimate goal is that they want voluntary transparency from services so that users know how data is being used, and this may lead users to change their behavior. The next questioner asked whether someone shoulder surfing who noticed ads targeted to another user, could exploit information about the user? Mathias responded that this is indeed a real threat, but more transparency is needed. The final three questions concerned the capabilities of the framework. Can it work over time series? Is it possible to have inputs that are more complex than a single word? Did they plan to extend the framework to work across multiple services? The answer to all three questions was yes.

Mass Pwnage

Summarized by Qi Alfred Chen (alfchen@umich.edu)

An Internet-Wide View of Internet-Wide Scanning

Zakir Durumeric, Michael Bailey, and J. Alex Halderman, University of Michigan

Zakir Durumeric first talked about the popularity of Internet-wide scans after releasing ZMap last year, and motivated the work by posing the questions, who is using ZMap and what is the security impact from these fast scanning tools? To answer these questions, the authors collected data from a large darknet during 2013 to 2014, and fingerprinted scanners such as ZMap. In the analysis, they found increasing amounts of scanning activity. A large portion of this activity was targeted at vulnerable services. They also characterized the scanning sources. Surprisingly, they found that instead of botnets, most of the scans came from bullet-proof hosting providers or from China, and also many of them came from academic researchers.

After characterizing the scanning landscape, Durumeric chose three case studies to study the scans related to recent vulnerabilities in Linksys routers, OpenSSL, and NTP. For the Linksys backdoor, scanning activities started within 48 hours, and continued for at least two months. For the NTP DDoS attack vulnerability, nearly all probing traffic was part of large scans and was primarily from bullet-proof hosting providers. For the Heartbleed bug in OpenSSL, scan activity began less than 24 hours after the disclosure, and the volume doubled in the following week. The scanning origins were either bullet-proof hosting providers or from China.

Durumeric also talked about the results on the defensive mechanisms for the scans. Although scanning activity largely increased, only 0.05% of the IP space was inaccessible from their scanner, and only 208 organizations have requested exclusion of scanning. They also summarized the scan detection mechanism deployed by organizations. Durumeric concluded that large scans have become common, while the defenders remain slow in responding to these scans.

Following the talk, Bill Cheswick asked whether they would be posting the set of ASes responsible for scanning, as he loves the idea of shunning. Durumeric answered yes, and he added that they are listed in the paper. The session chair, Vern Paxson (UCB), asked how they could tell if scanning was widespread, and Durumeric answered that if they saw a normal distribution, they assumed it was widespread. Paxson then pointed out that some sites block IP addresses for short periods of time, and Durumeric replied that they scanned every day.

On the Feasibility of Large-Scale Infections of iOS Devices

Tielei Wang, Yeongjin Jang, Yizheng Chen, Simon Chung, Billy Lau, and Wenke Lee, Georgia Institute of Technology

Tielei Wang investigated the possibility of infecting iOS devices on a large scale from compromised computers in a botnet. In this work, Wang identified two previously unknown attacks that enable compromised host computers to deliver malwares such as the Jekyll app (presented by him in last year's USENIX Security) into an iOS device via USB or WiFi-based syncing. Leveraging this vulnerability, a measurement study was then conducted to estimate the population of iOS devices connected with compromised computers in botnets, and they found that 23% of bots in the study can be used to infect iOS devices.

The first attack managed to install Apple-signed malware into iOS devices. A major challenge is that each downloaded app is associated with an Apple ID and cannot run on a device that is bounded with another Apple ID. However, the authors found that this protection can be bypassed when using the iTunes syncing feature. With a man-in-the-middle attack idea, this vulnerability can be exploited from a remote computer's iTunes, opening the door to iOS devices infected remotely with malwares such as the Jekyll app.

The second attack exploited the device-provisioning process, which is designed for testing purposes. Wang's work found that provisioning profiles can be installed, enabling the installation or removal of attacker-signed apps from a compromised computer via USB. Besides injecting apps, Wang also found that with a USB connection, a host computer can steal apps' sensitive files, such as cookies, which are well protected by sandbox-based isolation between apps but can be accessed easily from the host computer.

After demonstrating the threat from host computers to iOS devices, Wang used a measurement study to determine the scale of possible infection from compromised computers in the botnet to iOS devices. To get a lower bound of the percentage of the possible infected iOS devices, Wang analyzed a DNS query data set of a botnet to find out the number of iOS users using iTunes on the compromised computers with Windows systems. The analysis results showed that at least 23.70% of the iOS devices can be infected, and for a large botnet in the data set, the infection range could involve 13 cities.

Yossi Oren (Columbia) asked about a limitation of the attack of injecting attacker-signed apps, which requires an iOS developer license and can only provision 100 iOS devices. Wang replied that the Enterprise iOS Developer License can be used to circumvent that. He was also asked about whether the attack of injecting Apple-signed apps depends on iTunes, and he replied that using iTunes is not necessary since there are other third-party tools that can be used to do syncing.

A Large-Scale Analysis of the Security of Embedded Firmwares

Andrei Costin, Jonas Zaddach, Aurélien Francillon, and Davide Balzarotti, Eurecom

Andrei Costin motivated the work by talking about many insecure embedded systems such as routers, printers, and cars, and described the challenges of making the analysis of embedded firmware security large scale, including, for example, the heterogeneity of hardware, users, etc.; firmware data set building; and firmware identification and analysis.

To collect firmware data sets on a large scale, the authors used a Web crawler to automatically download files online and set up a Web site for users to submit firmware images. Identifying firmwares requires manual effort and remained a challenge. To unpack firmware images and identify custom formats, they extended BAT (Binary Analysis Toolkit) with a range of additional plugins. To enhance the scalability for unpacking and file carving, which are very CPU-intensive, the system uses a cloud computing platform, where the analysis tasks were distributed to several worker nodes. The analysis on the unpacked firmware images includes correlation, clustering, and data enrichment such as version banners and keywords.

Costin demonstrated their system using some examples. The first example was about the correlation engine for finding similarity between firmware images. He showed that by using fuzzy-hashes, the vulnerability propagation can be studied. The second example concerned private RSA keys stored in the firmware images. Using the RSA key correlation and vulnerability propagation, the private key providing access to the Web interface of some CCTV cameras was found to be reused across many firmware images of the same brand, affecting 30,000 online IP addresses. They also found that the vulnerable components of these CCTV cameras are shared with CCTV cameras from another vendor. Costin concluded with a summary of their results, which include 38 new vulnerabilities correlated to 140,000 online devices.

Cynthia Irvine (Naval Postgraduate School) asked about whether they worked with vendors to make the firmwares more secure. Costin replied that they had disclosed their findings to vendors and communicated with them. He added that they had a Web site and made their data, including the firmware images and the analysis results, public. Costin was also asked about whether there is any way to characterize the vulnerabilities: for example, to identify whether the vulnerability is from the firmware itself or a third party. He answered that currently identifying firmware is hard to automate and is error prone: for example, finding the version number is usually hard due to diverse firmware file formats. He was then asked about how to judge the coverage of their firmware data set since their sources are online crawling and individual submissions. He replied that currently it is hard to build a representative data set since the embedded systems have a very heterogeneous environment.

Exit from Hell? Reducing the Impact of Amplification DDoS Attacks

Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz, Ruhr-University Bochum

Christian Rossow presented work on an Internet-wide study of UDP-based amplification attacks, the authors' attempts to mitigate the attacks, along with discussions on potential next step attacks and root causes. Rossow started by showing the scanning results to study the amplifier landscape. They found that the number of amplifiers remained constant throughout the study, and multiple vulnerabilities simultaneously existed on many systems. They then fingerprinted the devices and found that NTP and SNMP amplifiers largely run on routers, while the majority of NetBIOS amplifiers run on desktop computers. They also found that most protocols had a high rate of amplifier churn, but the NTP protocol only had a negligible rate of churn since NTP amplifiers are usually assigned static IP addresses.

Considering that NTP is the worst among all known vulnerable protocols, they then studied potential mitigations towards NTP amplifiers. Rossow showed a timeline graph to demonstrate their success in the remediation process of notifying the administrators. After releasing articles about the NTP attack, updating the list of potential amplifiers, and weekly notifications, they found the number of NTP amplifiers dropped by 92.4% with an ongoing decrease. Although this campaign was shown to be very effective, they still experienced difficulty in reaching out to the providers.

After showing the influence on the amplifier landscape, they further thought about a potential next step for attackers who may turn to exploiting the TCP protocol for amplification attacks. They studied retransmissions during the TCP protocol handshake and found that the SYN/ACK segments will be sent repeatedly without being stopped by RST responses, thus achieving the effect of overloading the capacity of the victim's network. Evaluated on HTTP, Telnet, and CUPS, this amplification attack was found to enable an amplification factor of six or higher for most of the reachable hosts.

Finally, they studied the root cause for amplification attacks: IP address spoofing. A remote spoofer test was designed based on DNS, without the need for individuals running manual or tool-based tests. Using this remote test method, they identified more spoofing-enabled ASes than previous work.

Cynthia Irvine asked about their future plans. Rossow answered that they want to focus on identifying the source of the amplification traffic, even though it is spoofed. Someone asked whether they used ZMap in their Internet-wide scanning of UDP-based amplifiers. Rossow answered no and added that they developed their own scanning tool. Someone asked which TCP protocols could be exploited for amplification attacks, and Rossow answered that, for example, HTTP Telnet, and FTP can be exploited. The questioner asked whether these TCP-based amplification attacks were real. Rossow replied that they were real and reproducible.

Privacy Enhancing Technology

Summarized by David Adrian (davadria@umich.edu)

Never Been KIST: Tor's Congestion Management Blossoms with Kernel-Informed Socket Transport

Rob Jansen, US Naval Research Laboratory; John Geddes, University of Minnesota; Chris Wacek and Micah Sherr, Georgetown University; Paul Syverson, US Naval Research Laboratory

There is a large body of work showing that Tor is slow, but little work has been done to explain why or to locate where the congestion occurs in the Tor network and client. Rob Jansen presented work that measures where congestion occurs both in the client and in the network in order to determine and help eliminate the root causes of congestion.

By instrumenting the Tor client, the authors were able to measure the amount of delay that Tor packets spent in kernel-level and application-level buffers. After enhancing the Shadow Tor network-simulator to provide more TCP-level data and creating a simulated network of over 3000 Tor nodes, the largest simulated network to date, they were able to determine that congestion almost exclusively occurs in outbound kernel buffers.

To reduce the congestion, the authors reimplemented the Tor client to more efficiently schedule circuits across multiple sockets, and to keep Tor packets in application-level buffers in cases where the send call would cause the packet to be added to a queue rather than being flushed to the wire. By keeping the packet in Tor level-buffers, Tor is able to make more educated decisions about which socket to use. Jansen referred to their improvements as KIST (Kernel-Informed Socket Transport).

KIST resulted in less kernel-level congestion and more Tor-level congestion. However, this produced a net decrease in latency. This latency decrease does make certain attacks published by Hopper et al. more effective, but the authors believe this is purely due to the fact that latency is lower and is not a direct flaw of KIST.

Someone asked whether they had measured what happens when some Tor clients have the authors' changes and some do not. Jansen stated they had not yet explored this area, and possibly will attempt to measure this before KIST is fully rolled out.

Effective Attacks and Provable Defenses for Website Fingerprinting

Tao Wang, University of Waterloo; Xiang Cai, Rishab Nithyanand, and Rob Johnson, Stony Brook University; Ian Goldberg, University of Waterloo

Tao Wang explained that the goal of Web site fingerprinting is to determine from Tor network flows which Web sites a particular Tor user is visiting by fingerprinting the traffic of a specific Web site. Wang presented a new, more effective fingerprinting attack and a provable defense against it. However, the provable defense has a large performance impact. To explain the work, Wang used comical slides featuring a small fuzzy character who was using Tor and being attacked by a red blob.

The attack is machine-learning-based and uses the k-nearest neighbors algorithm with a specially trained distance measurement that weights various features visible in encrypted Tor flows, such as packet size and timing. With the classifier trained for a variety of popular Web sites, when used against a Tor client visiting both fingerprinted and non-fingerprinted Web sites, the attack is able to achieve an 85% true positive rate with a 0.6% false positive rate.

To defend against the attack, the authors implemented a provably secure defense that uses the shortest-common supersequence to enforce that all packet flows appear identical. However, while this method is provably secure, it requires at least a 60% bandwidth overhead. A non-provably-secure defense can use as little as 6% bandwidth overhead, but can still be broken with enough data, effort, and time.

TapDance: End-to-Middle Anticensorship without Flow Blocking

Eric Wustrow, Colleen M. Swanson, and J. Alex Halderman, University of Michigan

Eric Wustrow explained that since 2011, there have been several different papers that describe methods for performing anticensorship using end-to-middle proxies, such as Decoy Routing, Telex, and Cirripede. However, when the authors of this paper and Telex approached ISPs to deploy these end-to-middle proxies, the ISPs opposed the inline flow-blocking element—the part of the proxy that blocked traffic to the decoy host and redirected it to the censored host.

TapDance is an end-to-middle proxy that only needs a passive tap and the ability to inject packets, in place of the flow-blocking element. To achieve this, the authors used an insight gained from a careful reading of the HTTP and TCP specification. By sending an incomplete HTTP request to the decoy server that is missing the second \r\n, the decoy server will ignore the rest of the incoming packets that are intended for the censored host.

To indicate to the TapDance station that the incomplete request is in fact a TapDance request, the authors use a steganographic channel inside of the TLS ciphertext. For stream ciphers, the authors leverage the fact that flipping a bit on the input flips the corresponding bit on the output. Using this channel, the authors can encode a message encrypted with a TapDance station's public key.

Once the connection has been established, the censored client can send requests for censored content to the decoy server that are then decrypted by the TapDance station and proxied via an uncensored network to the rest of the Internet. The TapDance station then spoofs a response from the decoy server with the censored content.

Peter Honeyman expressed his surprise that any ISPs were willing to deploy such an anti-censorship tool, with or without flow-blocking. Wustrow replied that the amenable ISPs tended to be Tier 2 ISPs with a research focus, rather than a large consumer ISP such as Comcast.

A Bayesian Approach to Privacy Enforcement in Smartphones

Omer Tripp, IBM Research USA; Julia Rubin, IBM Research Israel

Omer Tripp described a tool designed to detect privacy leaks in non-malicious Android applications, such as applications that send user data to third-party advertising companies. Their approach uses a Bayesian classifier and assumes data is transferred using any of several common encodings, such as hex, base64, and JSON.

The authors implemented their system in a tool called BayesDroid. When applied to 54 applications from Google Play, the authors were able to find 27 new privacy violations with only one false positive. These results were more accurate with fewer false positives than TaintDroid, the current state-of-the-art taint tracking tool for Android privacy analysis.

Someone asked whether their small sample size might have led to overfitting. Tripp replied that they did not believe so, and that they were releasing their code with the hopes that third parties might use it and provide feedback and more training data.

Crime and Pun.../Measure-ment

Summarized by Andrei Costin (costin@eurecom.fr)

The Long "Taile" of Typosquatting Domain Names

Janos Szurdi, Carnegie Mellon University; Balazs Kocso and Gabor Cseh, Budapest University of Technology and Economics; Jonathan Spring, Carnegie Mellon University; Mark Felegyhazi, Budapest University of Technology and Economics; Chris Kanich, University of Illinois at Chicago

Janos Szurdi started his talk by introducing "typosquatting" domains. These are domain names that differ from domain names of known or established brands by a small difference: for example, "google.com" versus "googl.com". Users usually end up on these domains by having a typo in the intended domain names, hence the term "typosquatting." Janos mentioned there around 56 million potential typosquatting domains in the whole .com range. And while most previous research focused at most on the top 200,000 Alexa sites to look for typosquatting intelligence, they performed a comprehensive study across the entire .com domain distribution to gather a more complete understanding of the typosquatting phenomenon.

Janos presented a case-study example based on PNCBank, which is a Top 10 US bank and is hosted at pncbank.com. There exists pncbnk.com, which runs advertising, pncban.com, which runs a survey that turns out to ask for credit card and personal details at the end, and, finally, pvbank.com, which claims to be a bank in India and even has a disclaimer that states the site is not a typosquatting domain! The case of pncbank.com, Janos says, could have been missed by previous researchers due to its lower Alexa ranking. Moreover, only 2% of the typosquatted domains were found to be associated with their related brand owners (either by domain owner directly or via brand protection services). The other 98% of the cases were parked serving ads, and a minority of them were associated with either competitors' domains, phishing, or malicious pages.

In their methodology, Janos said, they used 1-Levenshtein distance to generate typosquatting based on real and valid domains. Thus they used: deletion of a character, addition of a character, substitution of a character, switching of two adjacent characters, and appending the "www" prefix to the TLD domain name.

Janos noted that some mistyped domains are "true typosquatting" domain cases, while some are not and are "incidentally typosquatting" (i.e., not a true typosquatting case) similar to the original brand only incidentally. To find a "true typosquatting" case, several pieces of information are used, such as DNS and WHOIS records, the domain's content, and redirection chains. However, there was a need to develop heuristic rules to decide whether a domain is an instance of "true typosquatting" or "incidental typosquatting." The challenge in developing such heuristics, however, is the lack of ground truth.

Based on Alexa, the authors took four sets of .com domains to form the ground truth. Sets were as follows: Alexa rank 1–10,000, Alexa rank 10,000–250,000, Alexa rank 250,000–1,000,000, and a set of randomly chosen .com domains. For each set, they performed typo-generation (as presented above) and selected 100 random domains from each typo-generated set. Finally, they compared their results with the other two previous works. For the top domains set, they found that most typo domains are "true typosquatting" domains, and all three classifiers performed with similar accuracy. For the other three less popular domain sets, the other two classifiers decreased in performance, while Janos' classifier kept almost the same performance as for the top domains set. In addition, for the set Alexa rank 250,000–1,000,000, and the set of randomly chosen .com domains, less than 70% were found to be typo-domains.

Janos highlighted the following trends in the registration time of typosquatting domains. There is a continuous battle between brand protections and typosquatters. Obviously, over time typosquatters cannot register popular domains. In addition to this, more than 70% of typosquatting domains are registered more than one year, which means domains are truly desired/desirable.

Janos pointed out that 25% of typosquatting domains were registered with one registrar, and another 15% with six registrars. One solution, therefore, would be for ICANN and VeriSign to force on such registrars additional checks to prevent typosquatting. However, there is no incentive to do so since it will decrease the revenue of both ICANN, VeriSign, and the registrars.

Janos concluded that the typosquatting phenomenon is widespread and is also targeting less-popular domains as well. Another conclusion drawn was that the number of popular typosquatting domains increases over time.

Jeffrey Goldberg (AgileBits) asked whether some substitutions are more common than others for genuine typosquatting. Janos mentioned that deletion and so-called "fat finger" typos were the most common substitutions. Someone asked whether one big group was behind the typosquatting, or hundreds of groups, and what indicators could and would be used to answer such a question. Janos clarified that there are few groups massively typosquatting, as well as some other smaller groups for whom typosquatting was not the main business. As for differentiating between the groups, authors used DNS, WHOIS, and contents of the site. Janos was asked whether typosquatting domains use HTTPS and whether HTTPS is more or less common among the typosquatting domains. Janos said they unfortunately didn't look at HTTPS and this is a good point to look at in the future. The final question was whether the auto-correction features (e.g., in browsers and search engines) would help to avoid landing on typosquatting domains. Janos said they didn't study this direction, but he thought it would help a lot.

Understanding the Dark Side of Domain Parking

Sumayah Alrwais, Indiana University Bloomington and King Saud University; Kan Yuan, Indiana University Bloomington; Eihal Alowaisheq, Indiana University Bloomington and King Saud University; Zhou Li, Indiana University Bloomington and RSA Laboratories; XiaoFeng Wang, Indiana University Bloomington

Sumayah Alrwais began by introducing "parked domains." These are "unused" domains that receive a large amount of traffic and that usually register with advertising networks for high revenues. A common choice for running such domains are domain parking services (DPS). These are usually running various advertising network campaigns and are addressed to various markets, advertising types, and revenues.

Sumayah presented several examples depicting different parked domain scenarios. One is "education-guide.org," registered with a DPS and running multiple advertising networks (Google AdSense, advertise.com, Bing Ads). Another is "city-cars.net," which sells traffic for specific search keywords. This type of traffic is sold via keyword-related companies and traffic systems such as DNTX.com. A final example is "expeedeea.com," which is an obvious typo-domain for "expedia.com" and is registered with a brand protection service.

Sumayah briefly explained the ways the domains are parked and monetized. One option is to simply redirect traffic via HTTP 302 redirects. Another option, the most common one, is to set the name servers (NS) of the domain to those of the DPS, such as ns1.sedo.com. In this case, it gives the DPS complete control over the traffic, hence the best content, monetization, and redirection strategies.

Sumayah explained that they wanted to study the legitimacy of the DPS operation. Are they honestly reporting revenues to all the parties using their services? Are the advertisers receiving real clicks, real traffic for purchased keywords, and are the landing pages safe and free of malware? She explained there are inherent difficulties while investigating DPS. One challenge is the complex system of DNS and registrations, in addition to the many parties involved (DPS, Web users, domain owners, advertisers, ads-networks, proxies). Another challenge is that every actor has only a partial view of the entire monetization chain.

Sumayah presented their approach, which basically is an attempt to capture the end-to-end view of the monetization chain by using multiple actors to enter the domain parking monetization ecosystem. One actor registered domains and parked them with DPS (as domain owner). Another set up sites for "fake" advertisers (as advertisers) and bought advertising and traffic keywords. Finally, as Web users, they instrumented Web crawlers to search for millions of parked domains. Some of the crawled domains were operated by the authors themselves. She explained that this activity proved non-trivial with many challenges. From the advertiser point of view, they had to impersonate multiple user-agents of Web users. As traffic purchasers, they had to purchase keywords similar to domain names owned by the authors. The whole experiment consisted of capturing and analyzing 1.2 million chains, with 1015 end-to-end seed chains used as ground truth. The authors then applied fingerprinting on the monetization chains by using click/traffic stamps inside URL patterns and IP addresses that identified a particular monetization party.

Sumayah described the many frauds they discovered. The authors found click fraud in 45.7% of chains and at least 709 fraudulent clicks. In 38% of chains, they also detected "traffic spam" fraud, by receiving purchased traffic from completely unrelated domain names (e.g., for keyword "coupon"). They found malware distribution (via social engineering or drive-by downloads) in 2% of the chains. Finally, they detected "traffic stealing" fraud, which means not reporting the whole revenue to the domain owners. In this particular case, the authors paid 10 cents as traffic buyers. However, the assumed revenue did not propagate to the parking domain provider or to the real domain owner (who parked the domain, the authors in this experiment).

This resulted in the fact that traffic buyer got fully billed while the real domain owner got no revenue at all.

In conclusion, Sumayah said they estimated that more than 40% of the revenue of PDS is illicit. This, in the authors' view, calls for immediate regulation of domain parking businesses. Until then, there are several mitigations possible. One is that search advertising networks should label publishers with categories. Another is the integrity check and protection on the traffic buyers' side.

Andrei Costin (Eurecom) asked why Google entered the DPS market and why it left shortly thereafter. Sumayah thinks Google entered the market to tap into the search-keywords pool. However, Google got sued and most probably it was a big legal pain, which meant that it was easier to exit the market than to deal with all the legal and unregulated aspects. A follow-up question by Andrei was whether Sumayah could provide some advice regarding taxes, IRS, and legal aspects of their research. Sumayah encouraged researchers to always speak first to their legal department as this would surely solve many of the issues upfront, at least it did for them. Someone asked about the exact mechanism DPS created for click frauds. Sumayah said the fraudulent DPS had no hidden iFrames but were simply redirecting the click to a referral link. This resulted in a valid click from the advertiser point of view. Someone asked whether they reported the frauds to the FTC, and if so was the FTC interested in the reports. Sumayah said they did not report their results to the FTC.

Towards Detecting Anomalous User Behavior in Online Social Networks

Bimal Viswanath, M. Ahmad Bashir, Max Planck Institute for Software Systems (MPI-SWS); Mark Crovella, Boston University; Saikat Guha, Microsoft Research; Krishna P. Gummadi, Max Planck Institute for Software Systems (MPI-SWS); Balachander Krishnamurthy, AT&T Labs–Research; Alan Mislove, Northeastern University

Bimal Viswanath started by presenting service abuse as an anomalous user behavior problem. Service abuse is a serious problem today, where, for example, for less than $20 one can buy 5,000 "Likes." This phenomenon has especially negative effects for social advertising services.

The goal of Bimal's team is to detect and limit service abuse. This includes detecting the identity of such users and nullifying their identities or accounts. This is challenging, however, because of the adversarial cycle, where the "attacker" (i.e., abuser) mutates and always has an upper hand over the system. It is well known that attackers mutate by using fake accounts or compromised real accounts, or when real users collude with their identities to boost their ranking.

Bimal and his team suggested the approach of building an anomaly classifier. It is completely unsupervised and requires no training or labeled data. Additionally, it requires knowledge of attackers' strategies. Their approach contributes to the detection of "Like" spammers on social networks such as Facebook, regardless of spammers' strategies. Bimal indicated that for the approach to work, the classifier needs to learn patterns of normal user behavior. Hence, to evade detection the attackers need to act along with this learned user behavior, which limits and constrains the attackers in their actions.

Bimal introduced tables of "Normal vs. Anomalous" users. These are based not just on the number of categories and number of likes in each category, but also on an "inconsistent" small number of categories. He noted that behavior also changes over time, so they used PCA (Principal Component Analysis) to detect anomaly. Then he presented the capture of normal behavior patterns. To detect the few patterns of behavior that are dominant, Bimal said they used variance captured by each principal component from PCA. He confirmed this approach to work on Facebook, Yelp, and Twitter.

Bimal explained they trained the classifier on the behavior of "Like" activities of a random large sample of Facebook users. Subsequently, they tested the classifier on 3,200 black market accounts (bots, fake accounts), on 1,000 compromised accounts, on 900 colluding accounts, and on 1,200 normal accounts. Their classifier successfully flagged 99% of black-market accounts. He then explained "click spam" detection on Facebook. They set up a real advertisement targeting US users for a survey link. They also set up a bluff advertisement (i.e., an almost empty ad) that would be expected to get almost no clicks. The surprising result he mentioned was that both ads received a similar number of clicks and similar activity on the landing page!

Bimal concluded that service abuse is a huge problem in social networks today. He and his team proposed an unsupervised anomaly detection scheme. They then evaluated their technique on extensive ground-truth data of anomalous behavior. Finally, they applied their approach to detect click-spam in a social network advertising platform.

Damon McCoy (George-Mason) asked whether the technique's applicability domain is highly constrained by the patterns analyzed. Bimal replied that the technique is quite general and works as long as one can model the "normal behavior" of a user in a given system. Someone asked whether the adversaries they assumed have specialized in compromised accounts. Bimal explained that it would be hard to catch compromised users, but the behavior of such accounts after compromise makes them easier to catch with the proposed system. A follow-up question was on the number of "normal users" to be compromised in order to manipulate the technique. Bimal suggested that it would require compromising more than 30% of the "normal users." Algis Rudy (Google) asked about the threshold for noticing the anomalous behavior. Bimal explained that it largely depends on the operator, but it is a tunable parameter that can be trained for false-negative and false-positive ratios, and also depends on how much threshold the operator wants to tolerate.

Man vs. Machine: Practical Adversarial Detection of Malicious Crowdsourcing Workers

Gang Wang, University of California, Santa Barbara; Tianyi Wang, University of California, Santa Barbara, and Tsinghua University; Haitao Zheng and Ben Y. Zhao, University of California, Santa Barbara

Gang Wang began the talk by introducing machine learning (ML) in the context of security and detection of malicious users. ML is a proven tool for security applications. It's widely used for email spam detection, intrusion and malware detection, authentication, and identification of fraudulent accounts (Sybils). Despite its success, ML also has weaknesses. The key vulnerability is that statistical classes are derived from a fixed data set. Strong adversaries may become aware of ML and try to circumvent it.

Gang's team focused on crowdturfing, defined as malicious crowdsourcing, which is the act of hiring a large army of real users for malicious attacks or activities. For example, it is used to create fake reviews, fake testimonials, political campaigns, and CAPTCHA solving. An online crowdturfing system focuses on finding crowd workers for a customer. In China, ZBJ and SDH are the two biggest such systems with revenues of millions of USD per year.

Gang suggested that ML can be used against crowdturfing. One reason is that it can do more sophisticated modeling of user behavior. Another is that it's the perfect context in which to study adversarial ML, where there are highly adaptive users seeking evasion. In the case of adversarial ML, he suggested the existence of "evasion attacks" (i.e., workers/attackers evade classifiers) and "poison attacks" (i.e., workers/attackers alter training data).

Gang then presented their goal to develop defenses against crowdturfing targeting Weibo. Other goals are to understand the impact of adversarial countermeasures, that is, to understand which classifiers are more accurate than others and in which scenarios one classifier outperforms the others.

Their methodology was to gather training data and build/train classifiers afterwards. As their ground-truth data set, they used the two largest crowdturfing services targeting Weibo (ZBJ, SDH), totaling three years of data with more than 20,000 Weibo campaigns. In addition, they used 35 features to train the classifiers for crowdturfing.

The performance of 60% for random forests (RF) and 50% for decision trees (DT) indicates that it's possible to build an accurate classifier to detect crowdturfing workers. The follow-up research challenge was to detect workers trying to evade the classifier by mimicking a "normal user" and also to understand what knowledge is practically available to evaders.

Gang presented the set of evasion models they designed. One is "optimal evasion," which can be per-worker optimal or globally optimal. Another is the "practical evasion scenario," where the attacker does not know the classifier threshold boundary, but only knows estimated normal user statistics and can adopt those features that make an action look "normal." For optimal evasion attacks, results showed that 99% of workers could evade the classifier by changing five or fewer features. Although the practical evasion attack required more features to be changed, most classifiers were found vulnerable.

Gang then discussed the poisoning attack, which is executed during the classifier training phase by crowdturfing service admins highly motivated to protect their Web sites and workers. These attacks use tampering of the training data of normal users to manipulate model training. One way to do this is to inject mislabeled samples to training data, which results in a wrong classifier. Another way is to alter worker behavior uniformly by enforcing system policies, hence making it hard for the classifier to separate those two subsets. For example, admins can force a predefined time-out for workers before taking another task, so by delaying workers' tasks the service admin preserves the worker. Gang underlined the effectiveness of poisoning attacks, especially where more accurate classifiers are more vulnerable. For example, 10% of poisoned samples boost false positives by 5%.

Gang concluded with some key observations. One is that accurate ML classifiers can be highly vulnerable. Another is that no single classifier excels in all the attack scenarios. As future work, Gang mentioned the improvement of the ML classifiers' robustness.

Someone asked whether selection of the attributes/features is important for building the model. Gang explained that in their experience all features are useful, but some are easier to modify. He mentioned that it would be interesting in the future to look at features versus the cost of their modification. Kurt Thomas (Google) mentioned last year's semi-supervised study and asked which was better for detecting crowdturfing. Gang explained that last-year's semi-supervised approach was mainly aimed at detecting Sybil attacks in social networks and assumed that the baseline majority of users were stable, good-faith users. For crowdturfing, in particular, the adversaries have changed: They are real users, and it's easier for them to adapt, and hence they require supervised learning. However, unsupervised learning for crowdturfing looks very prospective as well.

Work-in-Progress Reports

The summary of this session is available online as an electronic supplement: www.usenix.org/login/dec14.

Forensics

Summarized by Grant Ho (grantho14@cs.stanford.edu)

DSCRETE: Automatic Rendering of Forensic Information from Memory Images via Application Logic Reuse

Brendan Saltaformaggio, Zhongshu Gu, Xiangyu Zhang, and Dongyan Xu, Purdue University

Awarded Best Student Paper!

Brendan Saltaformaggio presented work on automatically reconstructing file content from memory images (snapshots of a system's volatile memory). While existing forensics techniques can recover many of the raw data structures from memory images via signature scanning, significant work still remains in order to interpret or extract meaningful content from the raw data structures; Saltaformaggio dubs this the "data structure content reverse engineering" problem, and the core contribution of this paper is a technique and system that tackles this problem. Their main idea is that the applications where data structures are found often contain the rendering and formatting logic needed to generate human-understandable content from the data structures; thus, DSCRETE attempts to identify and reuse these rendering/formatting functions (collectively referred to as the "P function") in a program's binary to automatically extract human-understandable output from memory images.

First, an investigator feeds an application binary (e.g., the PDF viewer on the machine under analysis) to DSCRETE. Next, DSCRETE automatically discovers rendering/printing functions in the binary by using slicing. A list of candidate entry points to the P function (the first function that will call the correct rendering functions to generate understandable content) is then created by searching for locations that take a heap pointer and for which all previously discovered rendering functions depend on (based on a dependency graph). The final, correct entry point is then found through "cross-state execution": repeatedly feeding data structures from the memory image to candidate entry points until a human-understandable file is generated; this correct entry point can be saved for future analysis.

Saltaformaggio concluded by noting that DSCRETE was highly effective at recovering many digital documents and presented several demos of DSCRETE recovering Gnome-paint images and PDF documents from memory images.

Thurston Dang (UC Berkeley) asked whether DSCRETE can handle files generated by interpreted applications or scripts. Saltaformaggio said that, currently, DSCRETE cannot handle such files, but he affirmed that this would be pursued in future work. Responding to another capabilities question from a researcher from the University of British Columbia, Saltaformaggio remarked that as long as ASLR was disabled on the investigator's machine, the memory image could come from a machine that used ASLR because of memory mapping techniques discussed in more detail in their paper. Finally, David Jacobson (Qualcomm) asked whether DSCRETE could handle memory images that were partially corrupt; Saltaformaggio replied that DSCRETE should perform fine so long as the kernel paging structures were intact, but even if they were corrupt, techniques such as DIMSUM from NDSS 2012 might be able to be used in conjunction with DSCRETE to effectively generate human-understandable content.

Cardinal Pill Testing of System Virtual Machines

Hao Shi, Abdulla Alwabel, and Jelena Mirkovic, USC Information Sciences Institute (ISI)

Hao Shi presented his work on systematically discovering red pills that can detect whether a program is being executed in a VM or real/bare-metal machine; this was joint work with fellow researchers at the University of Southern California. Although several existing papers discuss how to build red pills, Shi's work is the first to systematically explore the entire space of the Intel instruction manual to detect semantic and computation differences between a VM and bare-metal machine.

Rather than use randomized testing, which lacks complete coverage, or symbolic execution, which cannot assess computational differences such as floating point arithmetic, Shi's group generated a red pill test case for all instructions in the Intel x86 manual. They then executed each of these red pill test cases (instruction operations) on a bare-metal machine, Bochs, QEMU using TCG, and QEMU using hardware-assisted virtualization; after executing the red pill tests, they compared the system states of their bare-metal machine against each of the VMs to determine whether the red pill elicited a difference between the VM and bare-metal machine. For each VM, Shi's work discovered over 7,000 red pills that could be used to distinguish the VM from a real-machine.

Concluding his presentation, Shi noted that their red pills emerged from two sources: incorrect VM implementations of the instructions or under-specified instructions that result in naturally ambiguous and different implementations of x86 instructions. Looking forward, Shi sketched a defense idea against red pills that instruments a honeypot/VM so that it takes in the list of red pills discovered by Shi and automatically executes/returns the correct and expected value of a real machine.

One researcher noted that Shi's presentation and paper mentioned that they tested several different versions of QEMU such as 1.3.1, 1.6.2, and 1.7.0, but neither the presentation nor paper included results for the later versions of QEMU (1.6.2 or 1.7.0); Shi affirmed that his paper did contain results for 1.6.2 and 1.7.0, despite challenges from the audience member that the results could not be found in the published, online version. Since Shi's group tested for red pills across different versions (i.e., newer versions) of QEMU, one audience member asked whether it looked like QEMU was improving the fidelity of its emulation of a real machine; sadly, Shi noted that based on their test results, it does not appear that QEMU is improving.

BareCloud: Bare-Metal Analysis-Based Evasive Malware Detection

Dhilung Kirat, Giovanni Vigna, and Christopher Kruegel, University of California, Santa Barbara

Dhilung Kirat presented work on detecting binaries that attempt to evade VM analysis. While many dynamic analysis techniques have been developed to analyze malware samples, these analysis techniques are often conducted inside of a virtual machine called a "honeypot" when used for malware analysis. Unfortunately, a number of techniques have been developed that allow binaries to detect whether their execution environment is a VM or real (bare-metal) machine. The major contribution of Kirat's work is "BareCloud," a bare-metal analysis system and a technique to leverage the bare-metal system's analysis to detect whether a binary sample exhibits cloaking/evasive behavior in a VM.

BareCloud sets up the bare-metal machine with a clean system snapshot and then initiates a binary sample by sending it over the network and removing all traces of in-guest automation and analysis tools before the malware sample executes. After allowing the sample to execute, BareCloud extracts a behavioral profile from the bare-metal machine, which is a tree-based structure of the disk changes (relative to the clean system snapshot) and network activity; in order to keep the bare-metal system completely free of in-guest, detectable monitoring tools, the information collected by the bare-metal analysis is limited to just disk and network activity. Next, BareCloud extracts similar behavioral profiles from common honeypots by loading the same clean snapshot onto Anubis, Ether, and Cuckoo Sandbox and executing the same binary sample in those environments. These tree-structured behavior profiles are then compared using hierarchical similarity to detect significant deviations (i.e., evasion) between the bare-metal machine and honeypots; a significant deviation is defined by an empirically derived threshold set by the researchers.

To evaluate BareCloud, Kirat's team ran BareCloud on a corpus of 110,000 samples from Anubis; the samples varied from minimal activity (very few system events and no network traffic) to high activity (thousands of system events and more than 10 packets set over the network). Overall, they found that BareCloud labeled approximately 6,000 samples as evasive malware.

Inquiring about these results, Grant Ho (UC Berkeley) asked whether Kirat's team had analyzed what the false-positive and false-negative rates were for their 6,000/110,000 result metric. Kirat responded that there was no ground truth labeling, so it wasn't possible to precisely determine how many of the 6,000 samples might be false positives and how many evasive malware samples remained undetected in the full set of samples. Additionally, Kirat was asked whether the execution environments (i.e., the operating system version, libraries, and applications like Java) were identical among all sandboxes and the bare-metal machine. Kirat said that during the BareCloud experiments, all execution environments were kept exactly the same to ensure that behavior profile differences truly resulted from a sample's behavior. Answering another question about identical environments, Kirat noted that Anubis's auto-click feature (which allows the honeypot to automatically interact with samples) was disabled because the bare-metal system cannot have the same auto-click behavior without introducing detectable, in-guest automation components.

Blanket Execution: Dynamic Similarity Testing of Program Binaries and Components

Manuel Egele, Maverick Woo, Peter Chapman, and David Brumley, Carnegie Mellon University

Manuel Egele presented his work on a dynamic analysis technique to determine whether two functions are similar given just their binary representation. Determining whether two functions are actually similar or different, given their binary forms, has several security applications such as identifying polymorphic malware or analyzing updates/patches for vulnerability findings. Their tool, "Blex," introduces a novel dynamic analysis execution technique (dubbed "blanket execution"), which greatly increases coverage of program logic.

Like many existing dynamic equivalence checkers, Blex starts by executing two unknown functions, *f* and *g*, in a fixed environment and storing the side effects of both executions. But in addition to their initial execution, the blanket execution process continues to execute *f* and *g* (and collect their side effects) by finding the first unexecuted instruction in *f* and *g* and executing them from that point; this repeated execution and side effect collection continues until all instructions in *f* and *g* have been executed at least once. From the full set of side effects for *f* and *g*, Blex outputs a similarity score for the two functions by computing a Jaccard index of the two sets.

To evaluate the efficacy of Blex, Egele's group collected the functions from the GNU coreutils package and ran Blex on these functions, when compiled at different optimization levels. While BinDiff slightly outperforms Blex when functions are compiled at similar optimization levels (e.g., –O0 vs. –O1), Blex is roughly twice as effective as BinDiff when matching functions at very different optimization levels (e.g., –O0 vs. –O3).

Egele was asked how Blex performed when functions and binaries were obfuscated, as typically seen in malware. Egele responded that enhancing Blex to handle obfuscated code will be future work; this work was simply the first step to see whether their blanket execution technique was even possible, which is why they evaluated Blex on obfuscated versions of popular binaries.

Invited Talk

Summarized by Rik Farrow (rik@usenix.org)

Information Security War Room

Sergey Bratus, Dartmouth; Felix (FX) Lindner, Recurity Labs

Sergey and FX filled in at very short notice for a speaker unable to attend and provided a lively presentation. Unfortunately, FX was in DEFCON mode, and much of the talk was R-rated for language. Nevertheless, what the two had to say was definitely interesting and relevant to our current security landscape.

The speakers really had four main points to make: (1) we do not currently have secure systems; (2) there is no product liability for the software industry; (3) instead of fixing the problem, some countries are trying to defend the status quo using treaties; and (4) LangSec principles explain a lot of the security issues in current software.

FX began by pointing out that you cannot buy a secure system today, and that just two market sectors have no product liability: software and illegal drugs. Both of these groups call their customers "users," a comparison that earned FX some laughs. FX then displayed a graph created by Veracode, a company that examines software binaries looking for exploitable input conditions. The two worst software sectors for security flaws were in customer support software and security software. This should not be a surprise since security software usually has the focus of being built out of protocol parsers, a topic they get to later. But just imagine, said FX, having a Intrusion Prevention System with all of its rules built into the operating system: you have 500 protocol parsers right in the line of sight of the attacker.

FX, who acts as a NATO advisor in the cybersecurity realm, said he has hope that the military will be the first sector that requires that software be backed with product liability guarantees. FX said that the military already takes eight years to procure software, and buggy and insecure software empires could receive a fatal blow if companies will not stand behind the software that they sell by accepting liability for failures.

FX shifted into a related theme, the attempt to create treaties to "fix" the software security problem. He used the analogy of long bows, which made it possible for peasants to kill the expensively armored nobility and resulted in said nobility creating rules of warfare in an attempt to outlaw this "unfairness." FX described a simply amazing example, the new code of conduct for the Internet, proposed by China, Russia, Tajikistan, and Uzbekistan (and other countries) at the 66th session of the United Nations. FX claimed that this is not because Russia and China feel threatened by cyberattacks, but instead they feel they are in a superior position.

But FX and Sergey feel that the Wassenaar Arrangement is a much bigger threat to security today. This treaty even makes research into tools that might be used in cyberwarfare illegal. For example, software designed to avoid detection by monitoring tools, for extraction of data or information, or to modify the standard execution path of a program, would be illegal to export. As an example, Sergey pointed out that debuggers and dynamic loaders both modify the execution path, although exceptions are made for these by Wassenaar. But tools like debuggers came out of a need to understand how software works (or doesn't work), and treaties like this one would ban research into potentially dual-use tools. The speakers, and several others, have written a position paper and call to action that can be found in the online version of the August 2014 issue of *;login:* about the potential effects of Wassenaar.

Both speakers called for creating textbook definitions for words relating to security, such as "exploit" and (better) "weaponized exploit." Having these terms defined by lawmakers or journalists may result in making much security research illegal in most countries.

Shifting into the final topic, Sergey explained that software insecurity has everything to do with attempting to solve a problem that is unsolvable. The Chomsky-Schützenberger hierarchy (1956) describes sets of parsers that are required to recognize the four classes of formal grammars. Only regular grammars can be proven correct, and the use of any other more complex grammars leads to parsers that cannot be proven correct. Parsers handle input to software, and that's exactly the point where exploitation occurs. Too bad the Wassenaar Arrangement doesn't make context-free, context-sensitive, and unrestricted grammars in parsers illegal, as that would actually do a lot to improve software security.

Sergey stated that they wished to patch the Postel Principle: Instead of being liberal in what is accepted in input, input must be matched exactly. FX commented that we needed Postel's Principle to get TCP/IP off the ground. But now we must play by the rules of adults. They both went on to list a series of recent exploits, all of which are triggered by the parsing of input, including OpenBSD's IPv4 bug in 2007, the chunk encoding bug that appeared first in Apache in 2003, then again in Nginx in 2013, goto fail in Mac OS, and Heartbleed in 2014.

They ended by suggesting that people check out the papers from the LangSec workshop at IEEE Security and Privacy 2014: http://spw14.langsec.org/.

Steve Bellovin (Columbia) agreed with the point about the need for input recognizers. He provided the example of a fascinating bug in the FTP recognizer, which used a YACC grammar to parse input and led directly to the security hole. Steve concluded that the wrong type of grammar was used for the task. Sergey agreed with Steve's point, and commented that we focus mostly on computation at the syntactic side, but not on the semantic side. FX joked that if they hadn't used YACC, but the typical approach of using case statements, they would have had hundreds of bugs.

Attacks and Transparency

Summarized by Brendan Saltaformaggio (bdsaltaformaggio@gmail.com)

On the Practical Exploitability of Dual EC in TLS Implementations

Stephen Checkoway, Johns Hopkins University; Matthew Fredrikson, University of Wisconsin—Madison; Ruben Niederhagen, Technische Universiteit Eindhoven; Adam Everspaugh, University of Wisconsin—Madison; Matthew Green, Johns Hopkins University; Tanja Lange, Technische Universiteit Eindhoven; Thomas Ristenpart, University of Wisconsin—Madison; Daniel J. Bernstein, Technische Universiteit Eindhoven and University of Illinois at Chicago; Jake Maskiewicz and Hovav Shacham, University of California, San Diego

Stephen Checkoway from Johns Hopkins University began this presentation with a central assumption: An attacker can generate the constants used in Dual EC. Based on this assumption, the research aimed to analyze the cost of such attacks against TLS implementations (such as that used in SSL libraries) that use NIST's Dual EC pseudo-random number generator. After reminding the audience that the NSA has gone to great lengths to standardize Dual EC implementations, Checkoway shifted to a brief overview of the operations of Dual EC. Checkoway used diagrams to show where the product of each iteration becomes the seed of the next pseudo-random number generation, and by leaking that seed, it's easy to derive the next output of Dual EC.

In this work, the authors analyzed four common TLS libraries: RSA BSAFE for Java and C/C++, Microsoft Secure Channel, and OpenSSL-FIPS. Checkoway detailed how their attack, assuming the back door introduced by knowing the Dual EC constants exists, could be implemented. Next, the effectiveness of this attack against each of the four common TLS libraries was shown. Finally, a live demo was shown where data taken from a TLS connection via tcpdump was decrypted in real time in just 3.5 seconds.

The presentation drew a question from Jeremy Epstein, who made it clear that he was from NSF and not NSA. He asked whether this work had been shown to Dickie George, an ex-NSA employee who said that breaking Dual EC was impossible. Checkoway answered that the work had not but that hopefully this dialog could occur in the future.

iSeeYou: Disabling the MacBook Webcam Indicator LED

Matthew Brocker and Stephen Checkoway, Johns Hopkins University

Matthew Brocker reminded the audience that, increasingly, everything around us (from cars to toasters) is being fitted with processors. The MacBook's iSight webcam is no exception. After obtaining an iSight camera from a 2008 MacBook, the research aimed to answer two questions: (1) Can the iSight's firmware disable the LED? and (2) Can the host system replace the iSight's firmware with malicious firmware?

Unfortunately, both challenges are possible. By manipulating the RESET register on the iSight, malicious firmware can disable the LED light regardless of the camera's state (on or standby). Even more disturbing is that the iSight's firmware can be replaced by a non-root process running on the host. Using these two vulnerabilities, the presentation concluded with a demo of videoing the audience with an iSight camera and proof-of-concept application. During video display, the speaker enabled and disabled the LED light.

The questions mainly focused on solutions for the attack. Andrew West (Verisign Labs) asked whether a fix other than mechanically binding the LED to the camera's operation exists (such as, a hybrid hardware-software solution). Broker's reply was very practical: While many solutions exist, the devices are already widely used and thus it will be difficult to push a solution out to so many vulnerable devices. Someone asked whether they had informed Apple of their results. Broker answered yes, but that the devices are already deployed. He also noted that the camera used in this work was from 2008 and that new cameras are different but may still be exploitable. Jeremy Epstein pointed out that the Sun4 workstation in 1994 had the same problem.

From the Aether to the Ethernet—Attacking the Internet Using Broadcast Digital Television

Yossef Oren and Angelos D. Keromytis, Columbia University

Yossef Oren gave a highly entertaining presentation of how HbbTVs suffer from a trio of security vulnerabilities. First, users have no control over the life cycle of applications running on the TV. Secondly, Web-origin policies are specified by the application itself. Finally, the RF-distributed code is allowed access to Internet resources. The presentation focused on one of multiple attacks presented in the paper: injecting attack code into a TV radio signal.

In this attack, Oren described how an attacker could intercept a standard TV radio signal. Attack code could be injected into this signal and then rebroadcast from an attacker-controlled antenna. Any HbbTVs picking up this signal could then be used to execute the attacker's code and access the Internet to download more code or to perform various wide-scale attacks.

Oren concluded with a number of countermeasures for each of the three HbbTV security problems. However, Oren also noted that enacting such countermeasures would incur cost to the HbbTV provider—making them unlikely to be adopted. One questioner asked whether regulators were concerned about such attacks. Oren responded that previous research had looked into the privacy implications of HbbTV and that he was aware of some concern within the European Union. Another wondered whether user credentials could be stolen in these attacks. Oren said if they had been previously cached, they could be stolen.

Security Analysis of a Full-Body Scanner

Keaton Mowery, University of California, San Diego; Eric Wustrow, University of Michigan; Tom Wypych, Corey Singleton, Chris Comfort, and Eric Rescorla, University of California, San Diego; Stephen Checkoway, Johns Hopkins University; J. Alex Halderman, University of Michigan; Hovav Shacham, University of California, San Diego

Keaton Mowery opened this talk with some background of how his lab obtained a Rapiscan Secure 1000 Full-Body Scanner—naturally, from eBay. Upon arrival, the researchers aimed to answer three questions: (1) Is the Secure 1000 radiologically safe? (2) What privacy safeguards exist? and (3) How effective is the Secure 1000 at detecting contraband? After a brief explanation of x-ray physics, Mowery explained how the Secure 1000's imaging equipment operates, and what results they obtained. The physics is important, because backscatter radiation is only created by less dense matter, such as flesh, as opposed to more dense material, like the metal in guns or knives, which does not produced backscatter, and appears black on scans.

On the questions of radiation safety, they found that the Secure 1000 emits safe levels of radiation. Thanks to a simple, modular software design, an attacker would need physical access to replace the system's ROM to over-irradiate a scan subject. The remainder of the talk focused on the Secure 1000's contraband detection capabilities (or lack thereof). Several slides presented side-by-side images of a scan subject concealing handguns and plastic explosives, all completely indistinguishable from unarmed scan subjects. The talk concluded by asking for more open evaluation of the full-body scanners in use by the TSA. Mowery suggested visiting https://radsec.org for more information.

Andrew Drew (Qualcomm) asked what tradeoff the TSA may be making to deploy working systems more quickly. Mowery acknowledged that the devices were not perfect but perhaps better than nothing. One questioner asked whether image processing may be able to detect hidden contraband that the human eye cannot. Mowery replied that this was not tested and that he doubted it would help. Later, Mowery was asked how physically obvious the hidden contraband was (since the presentation only showed x-ray images). He responded that it depends on how small the contraband is: A knife was possible to hide, but hiding a handgun was difficult to do inconspicuously.

ROP: Return of the %edi

Summarized by Ben Stock (ben.stock@fau.de)

ROP Is Still Dangerous: Breaking Modern Defenses

Nicholas Carlini and David Wagner, University of California, Berkeley

Nicholas Carlini pointed out that even enhanced versions of these countermeasures can be bypassed using only gadgets in simple coreutil tools like diff. He discussed the fact that all approaches that rely on lightweight Control-Flow Integrity suffer from the aforementioned issues and new defenses need to be proposed.

In the Q&A, an attendee asked whether their outlined attacks work on both x86 and x64. Carlini replied that the exploits discussed in their paper were targeting both these architectures and that there are not fundamental differences between the two.

Stitching the Gadgets: On the Ineffectiveness of Coarse-Grained Control-Flow Integrity Protection

Lucas Davi and Ahmad-Reza Sadeghi, Intel CRI-SC at Technische Universität Darmstadt; Daniel Lehmann, Technische Universität Darmstadt; Fabian Monrose, The University of North Carolina at Chapel Hill

Lucas Davi showed that even by combining the most strict rules that have been proposed over the last few years to a so-called ÜberCFI, kernel32.dll still carries enough long, call-preceded gadgets to be Turing-complete.

A question that arose was how hard the long NOPs were to find. Davi pointed out that while not all sequences that might be used are without any side effects, many other gadgets exist that reverse the side effects. Therefore, combining two gadgets of such characteristics again leads to a NOP gadget. Another question concerned the type of initial exploit step used by the exploits presented by Davi, to which he replied that they relied both on stack as well as heap overflows.

Size Does Matter: Why Using Gadget-Chain Length to Prevent Code-Reuse Attacks Is Hard

Enes Göktaş, Vrije Universiteit Amsterdam; Elias Athanasopoulos, FORTH-ICS; Michalis Polychronakis, Columbia University; Herbert Bos, Vrije Universiteit Amsterdam; Georgios Portokalidis, Stevens Institute of Technology

Enes Göktaş focused on breaking existing countermeasures but also on analyzing parameters for the existing approaches with which they would work. In doing so, their work found that it is possible to mitigate the effects of vulnerabilities by tuning the parameters for kBouncer (maximum length for a sequence to be seen as a gadget and number of gadgets used) for specific applications. Nevertheless, no generic parameters could be found that would neither cause false positives nor effectively stop the attacks. They pointed out that applications need to be analyzed in advance to determine parameters for the protection schemes to properly work.

Oxymoron: Making Fine-Grained Memory Randomization Practical by Allowing Code Sharing

Michael Backes, Saarland University and Max Planck Institute for Software Systems (MPI-SWS); Stefan Nürnberger, Saarland University

Stefan Nürnberger noted that although ASLR is a viable technique to make an attacker's life harder due to unguessable addresses, it can be bypassed if just one address from a library is somehow leaked (since the libraries are in memory en bloc). One approach to counter this is to spread out libraries across memory. This, however, impairs the sharing of a library between processes since relative calls to other library functions must be replaced with absolute ones (as there is no longer a correlation between the addresses of different functions). On a recent version of Ubuntu, this effectively leads to 1.4 GB being wasted for duplicate libraries when the OS is fully loaded.

Nürnberger proposes to solve this by using a combination of segmentation and lookup tables for all functions. Inside a library, a function can then place an indirect call to the *n*-th function in a table. The address of that table does not have to be known during compile time, but rather is set in the segment register at runtime. In doing so and in putting the lookup table in a memory region that is not normally accessible by code, the proposed solution provides randomization of the address space as well and allows for code sharing at the same time. In total, their approach has a runtime overhead of up to 3.5% as well as 13.5% file size overhead.

Nürnberger was asked if the approach would also work for other architectures, such as ARM. To his mind, the system would work but would require more instructions (as ARM is a RISC architecture). One potential attack was brought up by another questioner, namely scraping pointers to said library functions from memory. In this case, since the addresses need to be on the stack for the matching returns, the addresses of single library functions could be leaked. Someone asked about implementation of the mechanism. Nürnberger outlined that the approach can be easily built into a generic compiler like gcc. The final question concerned its feasibility on x64 systems. Nürnberger replied that while there are differences related to segmentation between x64 and x86, the approach would work as well; the only pitfall was the fact that the address of the lookup table could no longer be hidden (due to the differences in segmentation).

Safer Sign-Ons

Summarized by Venkatanathan Varadarajan (venkatv@cs.wics.edu)

Password Managers: Attacks and Defenses

David Silver, Suman Jana, and Dan Boneh, Stanford University; Eric Chen and Collin Jackson, Carnegie Mellon University

David Silver presented various vulnerabilities in many commercially available password managers like LastPass, 1Password, Keepass, etc. that aim to provide user convenience but often end up compromising security in previously unforeseen scenarios. Silver and his team particularly focused on the poorly named password manager feature called *automatic autofill*. This feature proactively fills any previously seen username, password pairs on Web sites without user interaction as opposed to manual autofill, which requires user interaction. It is not always safe to automatically autofill passwords. For example, autofilling passwords on a page that suspiciously fails to provide a valid SSL certificate or when the HTML form action URL changed between the time the password was saved and when it is used may not always be secure. Silver pointed out that some password managers automatically autofill in these scenarios.

Silver detailed one particular attack scenario where a malicious coffee shop owner providing free WiFi service could steal passwords from the password manager for a totally unrelated Internet activity. He showed a prerecorded demo video where an innocuous visit to an online pizza ordering service could seamlessly redirect to a totally unrelated Web site (e.g., AT&T or an online banking Web site) with malicious JavaScript code embedded. This action could trick the password manager into autofilling the credentials to the embedded malicious code, which could save the credentials on a remote server. The malicious code redirects the user back to the user-requested Web site, completing all this in milliseconds and remaining visually unobservable to the user. All password managers using the automatic autofill feature were vulnerable to this attack. Silver also discussed various defense mechanisms against these attacks. He and his team observed that disabling automatic autofill (i.e., manual autofill) is immune to these attacks. Particularly, doing manual autofill and submit is both secure and more convenient to users because it requires only one click. A better and more secure alternative to this is what he called *secure filling;* among other constraints JavaScript code was not allowed to read autofilled contents. They implemented a prototype on the Chromium browser that required only 50 lines of code.

Following the talk, David Wagner (UC Berkeley) observed that the secure password filling defense might have usability costs associated with it as there are benign Web sites that want JavaScript code to read autofilled contents. He asked whether the team looked at ways to reduce this cost by remembering whether a JavaScript code tried to read the autofilled contents while saving. Silver agreed with this suggestion, although he pointed out such a mechanism may not be able to always protect users from malicious JavaScript code. Jeffrey Goldberg (AgileBits) posed multiple questions. First, he asked Silver how frequently the action URLs change in reality. Silver pointed out that irrespective of the frequency, it is important to observe that the action URLs should be from the same origin (or domain). Second, Goldberg felt that the secure autofilling feature seemed to be rigorous in not allowing JS code to read autofilled content and asked how many benign Web sites were affected. Silver responded that they found only 10 of the top Alexa Web sites that used AJAX were affected. Thirdly, when asked about the impact of the work on commercial password managers, Silver mentioned a couple: 1Password now warns about autofilling when there is a SSL certificate validation failure, and LastPass stopped autofilling passwords in forms that are displayed in an iFrame.

The Emperor's New Password Manager: Security Analysis of Web-Based Password Managers

Zhiwei Li, Warren He, Devdatta Akhawe, and Dawn Song, University of California, Berkeley

Zhiwei Li started out by mentioning the importance of password managers and their popularity but questioned their security. Particularly, the authors identified four important security properties any password manager should provide: master account security, credentials database security, collaborator integrity, and unlinkability (a property of a password manager that does not allow tracking of a user across Web sites). Li went on to present four classes of vulnerabilities that their research uncovered that none of the password managers were immune to.

First, Li observed that all password managers provide a bookmarklet feature and these bookmarklets often run in an insecure environment, which may include running in the context of a malicious JavaScript (JS) code. Second, he presented a vulnerability where a Cross-Site Request Forgery (CSRF) would let an attacker choose an arbitrary one-time password (OTP) which could, in turn, be used to hijack the master account. Third, Li presented an example of another class of attack where sharing an asset/credential between users could result in unintended disclosure of credentials of the user who initiated the collaboration. Li pointed out that such vulnerabilities arise as a result of mistaking authentication for authorization. Fourth, Li showed a pre-recorded video of a phishing attack (under the class of User Interface vulnerabilities) where Li and his team where able to phish for LastPass's master credentials while remaining unnoticeable to the user. Li concluded the talk mentioning that there is no single solution that could solve all these damaging vulnerabilities, and it might take years for password managers to mature.

Jeffrey Goldberg (AgileBits) asked how the commercial password manager responded to their vulnerabilities. Li responded that some of them patched their code to defend against certain vulnerabilities but not for others. Goldberg then asked for Li's opinion on how single sign-on services compare to password managers. Li responded that similar vulnerabilities may still haunt such services, and both require further research to make them secure.

SpanDex: Secure Password Tracking for Android

Landon P. Cox, Peter Gilbert, Geoffrey Lawler, Valentin Pistol, and Ali Razeen, Bi Wu, and Sai Cheemalapati, Duke University

Landon Cox, on behalf of his students, presented their research project SpanDex, a tool that uses taint-tracking of passwords to hunt down inappropriate use of passwords and phishing attempts in mobile apps. Cox first showed various examples of phishing apps that steal user credentials used for the real counterparts, for example, the Wroba Android app that steals banking credentials.

Cox provided a simple introduction to taint-tracking where a variable/data-item that one wishes to track is tagged and followed using data-dependency and taint-transferring or propagation logic. Particularly, there are two flows in the taint-propagation logic: explicit flows that transfer the taint because of direct assignments operations, and implicit flows that transfer taint because of control flow or complex interactions between variables. Cox and his team identified the latter as significantly harder to track but essential to get good coverage and security guarantees. One of the challenges in taint-tracking in these implicit flows is that it often results in over-tainting. Cox pointed out various ways to avoid over-tainting implicit flows by weighting the taints differently. He used an example to motivate this observation: a tainted variable *s* used in a condition, s == 0, does not leak much information compared to an explicit flow tainting. Similar optimizations were used in making the taint-tracking faster and efficient. Cox and his team used a symbolic execution tool to do the taint-tracking and prototyped an implementation of SpanDex for checking Android applications.

Jeffrey Goldberg (AgileBits) asked what kind of passwords they look at in this work and did all of them follow power law. Landon Cox responded that all passwords were strings of characters and nothing else. Someone asked whether they looked at passwords that were processed locally since all the examples that were mentioned involved sending password on the wire. Cox replied that among 50 applications that they looked into none did local processing of passwords. Finally, someone asked why Cox and his team restricted their evaluation to uniform or Zipf distribution of passwords and not a frequency-based distribution. Cox responded by recalling that the password data set they used in the evaluation consisted of unique passwords, and a frequency-based distribution is not possible using that data set.

SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities

Yuchen Zhou and David Evans, University of Virginia

Yuchen Zhou first introduced the concept of single sign-on where there is an identity provider (like Facebook, Google) who maintains user credentials and a third-party integrator who wants to authenticate a user's identity. Although the SSO SDKs claim that integrators require little or no knowledge of security expertise, Zhou's research shows that this is often not true and might result in various vulnerabilities such as credential misuse. One instance of such a vulnerability is the misuse of an access token provided by the identity provider. A malicious user could reuse an access token provided for a particular application to access a different and completely unrelated application if the application fails to check the application ID provided in the access token. Zhou pointed out various real applications that were vulnerable to this attack and how trivially they expose these access tokens.

Zhou next presented SSOScan, which scans Web sites (integrators) for both credential misuse and credential leakage. SSOScan consists of three components: the Enroller, which automatically registers and logs into the Web page under test; the Oracle, which verifies successful enrollment and confirms session authentication; and the Vulnerability Tester, which tests for vulnerabilities using the newly registered account. Zhou discussed various challenges with automating the Enroller and Oracle. Zhou then presented the evaluation results on the top approximately 20k QuantCast sites: Out of 1.7k sites that use Facebook's SSO, 20.3% of them had at least one vulnerability. Zhou also pointed out that 12.1% misused credentials and 8.6% leaked credentials, with 2.3% of them having buggy implementation. When looking at the correlation between number of vulnerabilities and popularity of the Web site, Zhou found that the number of vulnerabilities found by SSOScan were the same irrespective of the popularity of the Web site. Zhou and his team also contacted many vendors about these vulnerabilities, and some responded with a fix. Zhou concluded by releasing the SSOScan as a service where vendors could check their Web site implementations for various vulnerabilities disclosed in this research (available at www.ssoscan.org).

Someone asked why they only chose 9.3% of 20k sites. Zhou responded that they randomly chose the Web sites and also restricted themselves to US sites since they were constrained by the language used in the sites. Another person inquired whether this service could be misused with malicious intent. Zhou replied that vulnerable Web sites were not publicly disclosed and to the best of their knowledge there seemed to be no such misuse. When asked whether changing APIs help to make misuse or leaking credentials harder for the developers, Zhou responded that developers often fail to read the documentation before implementing and was not sure whether changing APIs would help in such cases. Finally, someone asked whether any of the reported vulnerabilities were exploited in the wild. Zhou did not know of any such exploitation, but he believed that there were highly sensitive sites (e.g., match.com) that were still vulnerable and waiting to be exploited.

Passwords

Summarized by Andrei Costin (costin@eurecom.fr)

A Large-Scale Empirical Analysis of Chinese Web Passwords

Zhigong Li and Weili Han, Fudan University; Wenyuan Xu, Zhejiang University

Zhigong Li began his talk with the observation that when surfing the Web, a user often needs to register an account and that requires a password. Another observation was that password choice has strong geo-location influence. Zhigong attempted to answer the following two questions: Do Chinese Internet users have better passwords than others? How can one efficiently guess their passwords? He also observed that Chinese users form the biggest Internet group, with over 600 million netizens.

Zhigong mentioned they used leaked password databases from the top five Chinese Web sites as well as from Yahoo! and RockYou leaks. Their methodology was based on characters used, patterns, and their analysis. They found that most popular in both Chinese and English leaks were passwords 123456 and 123456789. However, digits are more common in Chinese passwords than in English ones. This is also because in Chinese the pronunciation of some digits is similar to letters. For example, pronouncing the number “520” sounds like “I love you” in Chinese.

Zhigong then presented their analysis on the resistance to guessing Chinese passwords. He explained they used “alpha work factor” analysis, which is the number of guesses required for a given success probability alpha. In many cases the alpha work factor can be very high. They found RockYou and Yahoo! passwords have higher work factor for alpha < 2.5.

Zhigong also explained that Chinese characters are input using pinyins, compared to simple characters in English or other western languages. About 25% of Chinese passwords contain pinyins. Interestingly, the top Chinese pinyin among Chinese passwords is “woaini,” which stands for “I love you.”

Zhigong suggested that dates also play an important role in password formation. In Chinese passwords most dates are formatted as YYYYMMDD, while dates in English passwords appear as MMDDYYYY. An additional finding is that dates in both Chinese and English passwords are put at the end.

David Wagner (University of California, Berkeley) asked whether it’s possible to compare probabilistic CFG (P-CFG) to the password cracking tools (simpler to use, etc.). Zhigong answered that cracking tools are dictionary based; P-CFG can analyze the structure of the password and can add some rules (e.g., dates), while cracking tools cannot add rules and hence are harder to use and less effective. Someone asked how dates have been distinguished from other random numbers in the analysis. Zhigong explained that there of course could be false positives. While the approach is simpler for eight-digit dates, for six-digit dates it can be trickier, so some six-digit dates were removed from the training.

Password Portfolios and the Finite-Effort User: Sustainably Managing Large Numbers of Accounts

Dinei Florêncio and Cormac Herley, Microsoft Research; Paul C. van Oorschot, Carleton University

Cormac Herley started at a fast pace presenting the following observation: Although everyone knows the basic rules, that passwords should be random and should not be reused across accounts, virtually nobody follows them.

He then presented a simple calculation showing that to remember 100 different five-character passwords for 100 different accounts would require the user to remember more than 4000 bits!

To reduce this burden, the user could weaken the password, which means reducing the lg(S). However, the computations show the amount of information is still way to high to remember (i.e., 524 bits), even if lg(S) is zero! The hypothesis Cormac advanced is that group and reuse are the only way out for normal people. He then mentioned that there were many ways to organize a password portfolio. For example, doubling the number of passwords more than halves the password strength, while stronger passwords force more password reuse.

Cormac emphasized that the question is not “are longer passwords stronger than shorter passwords?” nor “how does one generate secure random passwords or mnemonics?” From their research perspective, the question is “how does one minimize the password portfolio’s expected loss?” For example, setting the effort to infinity minimizes the probability of harm in case of loss of password. Cormac, however, highlights the fact that users also care about the effort. So the true question is “how does one minimize the portfolio’s expected loss + the user effort involved?”

Cormac explained that such a question makes the user think the right thing to do is to have some accounts that are weakly protected. For example, that “123456” is a top password in the RockYou database reveals that maybe those users chose to spend the password effort on something that matters more. However, passwords unfortunately reveal many other things, which means the risks are not independent across the accounts and not dependent only on the strength. In addition, the risk to the i-th account also depends on external factors such as malware and keyloggers.

Cormac provided the criteria for optimality (for loss, effort, and password policies). First is the division of things into *G* different groups. At this stage, how to optimally divide things into groups and how to choose division boundaries between groups are open questions. Second is that groups adjacent to each other should have similar weighted loss: for example, you shouldn't "put all your most important stuff into the same basket."

Cormac concluded that random and unique passwords are infeasible for large portfolios and that the user's interest is to minimize L(oss)+E(ffort) rather than just L(oss) over a portfolio. He also concluded that strategies that exclude reuse or exclude weak passwords are both suboptimal. The final conclusion from Cormac is that the best strategy in password grouping is: high-value accounts with strong passwords (low probability of compromise) and low-value accounts with lower password policies (high probability of compromise).

Cormac was asked about the needed effort to remember which password is in which accounts' group out of *G* groups. Cormac confirmed that such effort is required and referenced a formula in the white paper. In response to another question, Cormac pointed out that this model is a simplification and that there are many other factors that might need to be considered. For example, how to choose optimal *G* is not part of this research.

Telepathwords: Preventing Weak Passwords by Reading Users' Minds

Saranga Komanduri, Richard Shay, and Lorrie Faith Cranor, Carnegie Mellon University; Cormac Herley and Stuart Schechter, Microsoft Research

Saranga Komanduri introduced the authentication eco-system, which, in his model, comprises users (creating passwords), attackers (hijacking users' passwords), and admins (blocking attackers and protecting users). He also noted that admins create password policies to protect users. It is long known that simple password policies do not solve the issue. For example, Qwerty!123456 or Thisismypassword! adhere to policy but still can be viewed as weak. He also noted that in systems running Microsoft's AD, the three-class policies did not seem to have a notable effect in increasing the security of those systems.

Saranga presented his team goal as focusing on the weakest passwords. He also mentioned their contributions. One, they have shown that character requirements do not prevent weak passwords. Another, they detect weak passwords with guessability and real-time feedback.

Saranga went on to present their system, Telepathwords. It is similar to auto-complete, but its goal is to prevent the input of weak passwords. He underlined the fact that showing the user that a machine can guess the passwords is a good demonstration that attackers could do the same. He pointed out that password policies have not changed much since 1979, when the six-character password policy was proposed. At the same time, password strength meters are typically based on character requirements, and there is no consistency across different meters. Even though the "zxcvbn" open source meter, which estimates the entropy of passwords, was introduced to improve all the above, these meters still don't explain their scores.

Saranga explained they generated predictions using multiple models, including *N*-gram models, keyboard layout modes (e.g., similar to what some password cracking software does), repetitive passwords (e.g., abcabc), and interleaved strings (e.g., p*a*s*s*). The system can also be extended with more and better predictors. Then they created multiple policy conditions for a randomized controlled study. The study was designed as a hypothetical email scenario for password creation.

Saranga presented several policy metrics they used to assess the results. They used Weir+ guessability (i.e., finding single weakest password in the sample), zxcvbn entropy estimation (i.e., minimizing entropy in each sample set), probability conditions, usability metrics for creation of the password, and recalling of the password after several days of experiment.

Finally, Saranga presented some of the results. For example, dictionary-based policies are much better than their simple character counterparts. A real surprise is that 3class8 is not essentially stronger than basic8 with regards to guessability. Also, the policy did not affect the recall metric after 2–5 days. As for the creation time metric, Telepathwords policies took the longest time (90 seconds) compared to an average of 20–40 seconds for other policies.

Saranga concluded that character class policies had little to no effect in creating stronger passwords (i.e., less guessable ones). Additionally, dictionary policies' real-time feedback can help users create stronger passwords, but incurs a usability cost at creation time. Telepathwords was found to help users understand why their passwords were weak.

In response to a question, Saranga confirmed that Telepathwords are not harder to remember compared to random password generators according to their results, although they take a little bit longer to create and read the system's feedback. David Wagner (University of California, Berkeley) asked whether it's possible to download Telepathwords code and use on Web sites. Saranga said that Telepathwords is up and online, and that anyone can use it.

Towards Reliable Storage of 56-bit Secrets in Human Memory

Joseph Bonneau, Princeton University; Stuart Schechter, Microsoft Research

Stuart Schechter started with the observation that a user-chosen secret can never be provably hard to guess. Hence such a provably hard-to-guess secret has to be randomly generated by the system, for example, by a 56-bit secret key. There are multiple scenarios when a 56-bit strong key is required. One example is the master password for password managers. Another example is the access to organizations with a large number of users where, for example, weaker passwords cannot be filtered out and give attackers easier access to the organization.

Stuart pointed out that such an approach does not mandate that all Web sites now start using such unique 56-bit secrets, but that they be used only for critically important cases. Stuart suggested that we all use metaphors to explain problems, but the fact is that writing to the brain is harder than writing to the hard disk, and these metaphors can sometimes obscure reality.

Stuart then noted that our brains are designed to forget random data seen only once. He also noted that we have all learned through spaced repetition. We use repetition to learn important things, and we know it works great for learning. So the question Stuart and his team tried to address is whether humans could apply the same learning through spaced repetition to a 56-bit secret and how to apply this to remembering the secrets.

Stuart explained the setup of their experiment that tried to answer this question. The experiment involved subjects recruited via Mechanical Turk. The learning and recalling was camouflaged as a login to the system, which was presented to subjects as an experiment for something else (to avoid subject bias, suspicions, tricking). The 56-bit secret was presented for learning and recalling in three groups of four characters each. Some subjects had groups formed of random letters, while other subjects had groups formed of meaningful words. The system was designed close to reality, requiring the subjects to log in around 10 times a day, hence simulating an average workday in an enterprise.

Stuart then presented the results. Four subjects stopped learning codes for various and even funny reasons. On average, the subjects learned the entire secret at their 14th login attempt. For all subjects, learning the first group/code took most of the time. This could be for multiple reasons, such as not being used to the system or not understanding the system well enough at the beginning. Stuart presented the result that in this system there was only a 12% password forget rate registered compared to 26% in Telepathwords. Another finding was that recall rates decreased after more than two weeks. Finally, the recall rate for passwords grouped in word groups was 62% compared to 56% of the passwords grouped in random letters groups.

Andrei Costin (Eurecom) asked whether group interference across multiple 56-bit secrets (i.e., for multiple secrets, which group goes where and to which secret) was studied and how subjects responded to these interferences. Stuart explained that definitely interference is a potential direction for study, but they did not study this at present. He added that in this single 56-bit secret experiment, the results show that 2nd and 3rd group (letters, words) did not interfere inside this single 56-bit secret. He finally added that interference should not be a problem, since it is expected that a normal user should require only two or a maximum of three such 56-bit secrets during their lifetime.

Web Security: The Browser Strikes Back

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

Automatically Detecting Vulnerable Websites Before They Turn Malicious

Kyle Soska and Nicolas Christin, Carnegie Mellon University

Awarded Best Student Paper!

Kyle Soska presented a novel system that aims to predict which Web sites will become malicious in the future. There are many challenges in achieving such a difficult task. The target data set is the entire Web, a continuously growing data set of billions of Web pages. Moreover the data set is highly unbalanced, with many benign Web pages and a few malicious ones, the labels of the data set are incomplete, and there is no ground truth. Additionally, just predicting a Web site as potentially malicious in the future is not so useful on its own; webmasters need to be aware of the reasoning behind such a prediction so that they can react preventively. Lastly, the Web evolves over time; attacks change as new vulnerabilities are discovered, and the system should be able to react and adapt to these changes.

To cope with these challenges, the authors created a classifier based on C4.5 decision trees. With the use of blacklists and archive.org they created a data set of soon-to-be malicious Web sites and benign Web sites. The authors managed to isolate user-generated content from the visited pages with the use of composite importance, so that the system is able to focus only on the template-generated part of the Web pages, where vulnerabilities might exist. With a combination of dynamic and static features the system is able to achieve up to a 66% true positive rate with 17% false positives, showing this way that predicting malicious Web pages is possible.

Hulk: Eliciting Malicious Behavior in Browser Extensions

Alexandros Kapravelos, University of California, Santa Barbara; Chris Grier, University of California, Berkeley, and International Computer Science Institute; Neha Chachra, University of California, San Diego; Christopher Kruegel and Giovanni Vigna, University of California, Santa Barbara; Vern Paxson, University of California, Berkeley, and International Computer Science Institute

Alexandros Kapravelos focused on browser extensions: small HTML and JavaScript programs that modify and enhance the functionality of the browser. Alexandros showed that to compromise the browser, the attackers no longer need a 0-day exploit, but they can gain sufficient access in the user's system through a malicious extension. These extensions can inject more advertisements or perform affiliate fraud among other things.

To deal with this problem the authors developed Hulk, a system that dynamically analyzes and automatically detects malicious browser extensions. They introduced the notion of HoneyPages, which are dynamic pages that change on the fly to match what the currently analyzed extension is looking for in the content of a visited page. Hulk identifies malicious behavior by monitoring all aspects of the extension's execution. For example, Hulk will detect if the extension is preventing the user from uninstalling it or if it is stealing login credentials from forms. Hulk has analyzed 48k extensions and found 130 to be malicious and 4,712 of them suspicious. In the last part of the talk, Alexandros elaborated on the needed extension architecture changes that could either ease the analysis of extensions or prevent certain types of malicious extensions.

Precise Client-Side Protection against DOM-based Cross-Site Scripting

Ben Stock, University of Erlangen-Nuremberg; Sebastian Lekies, Tobias Mueller, Patrick Spiegel, and Martin Johns, SAP AG

Ben Stock focused on current defenses against DOM-based XSS attacks and how those can be circumvented automatically. By going over an extensive list of the current limitations of string-based XSS filters, the authors showed that the current state-of-the-art in DOM-based XSS can be evaded. Moreover, they implemented an engine that will automatically exploit 1,169 out of 1,602 real-world vulnerabilities despite the current defenses.

Ben made the simple, yet powerful, observation that client-side XSS filters use string comparison to approximate data flow, but this is unnecessary since it happens on the client side. Therefore, they propose a taint-enhanced JavaScript engine that tracks the flow of attacker-controlled data in a combination of taint-aware JavaScript and HTML parsers capable of detecting generation of code from tainted values. The new proposed method catches every single exploit, yielding no false negatives, and a 0.16% false positive rate for all analyzed documents. Moreover, the performance penalty introduced by the new defense mechanism was between 7–17%, with some optimizations applicable.

On the Effective Prevention of TLS Man-in-the-Middle Attacks in Web Applications

Nikolaos Karapanos and Srdjan Capkun, ETH Zürich

Nikos Karapanos focused on TLS man-in-the-middle attacks in Web applications. In the current state, server authentication can be circumvented by compromising a certificate authority, compromising the server's key (e.g., via the Heartbleed bug) or simply by letting the user click through the certificate warning on her own. Nikos elaborated on how TLS Channel IDs work, which is the current state-of-the-art for defending against MITM attacks. He then showed how one can circumvent TLS Channel IDs by proposing a new attack called man-in-the-middle-script-in-the-browser. This attack works not by impersonating the user to the server, but by injecting back to the user JavaScript code that will run in the context of the user's browser. Any communication with the server from the browser is now properly authenticated since it comes directly from the user's browser, but the code is controlled by the attacker.

To cope with this new type of attack, the authors propose Server Invariance with Strong Client Authentication (SISCA). This novel approach is based on the observation that the client needs to communicate with multiple entities for the MITM-SITB attack to work. By combining SISCA and TLS Channel IDs, Nikos showed that MITM attacks can be successfully prevented.

Poster Session

The summary of this session is available online as an electronic supplement: www.usenix.org/login/dec14.

Side Channels

Summarized by Qi Alfred Chen (alfchen@umich.edu)

Scheduler-Based Defenses against Cross-VM Side-Channels

Venkatanathan Varadarajan, Thomas Ristenpart, and Michael Swift, University of Wisconsin—Madison

Venkatanathan Varadarajan first introduced multi-tenancy in public clouds, which can benefit the utilization and reduce service cost but results in cross-VM attacks due to the difficulty of isolating resources. In their work, the authors targeted Prime+Probe cross-VM side channels that exploit per-core resource sharing: for example, the attack proposed by Zhang et al. in CCS '12, which demonstrated the possibility of extracting ElGamal secret keys. They found that for these attacks to succeed, quick preemption is required on the victim VM, which may be defended by limiting the frequency of VM interactions. This defense idea, which they called soft isolation, allows sharing with low overhead, and at the same time only needs simple changes to the hypervisor's CPU scheduler. After overviewing the high-level idea, Varadarajan introduced some background about the requirements along with the corresponding reasons for the availability of quick preemption in the cache-based side-channel attacks.

To achieve soft isolation, they proposed using the Minimum RunTime (MRT) guarantee, which is available in Xen and KVM. Varadarajan first presented the security evaluation of the MRT mechanism against cross-VM side channels. In a public cloud-like setting, using victims based on a simple model, they showed that under 1 ms MRT the side channel was not observable by the best known attacker. For ElGamal victims, the number of iterations per preemption was shown to increase dramatically to 32 under 0.5 ms MRT and to 68 under 1 ms MRT. This made those cache-based attacks very unlikely to succeed since they require multiple preemptions within one iteration for noise-reduction.

After demonstrating that soft isolation has the potential to defend against cache-based side-channel attacks, Varadarajan showed the performance overhead for normal applications when using MRT. Under 5 ms MRT, both interactive and batch workloads in the experiments had less than 7% overhead. Varadarajan also claimed that with 5 ms MRT and selective state cleansing (detailed in the paper), the overhead was negligible and no known attacks could work.

Varadarajan was asked whether this defense could be applied to other VM hypervisors besides Xen, and he replied that the minimum runtime support is not specific to Xen; for example, it also exists in Linux. He added that it is not particularly tied to a specific VM hypervisor, although there are still things to work out on other systems: for example, requiring performance analysis on other systems to achieve low overhead. Varadarajan was also asked how their defense compares to existing defenses, and he answered that previous defense methods mostly require dedicated hardware or else they sacrifice significant workload performance, while their defense is easier to deploy and also has low overhead under the current hypervisor settings.

Preventing Cryptographic Key Leakage in Cloud Virtual Machines

Erman Pattuk, Murat Kantarcioglu, Zhiqiang Lin, and Huseyin Ulusoy, The University of Texas at Dallas

Murat Kantarcioglu motivated their work by various security threats to the cloud VMs because of physical machine resource sharing: for example, many side-channel attacks have the potential of extracting cryptographic keys. To protect the secret key, their work proposes an idea of partitioning the keys into many shares using secret sharing and threshold cryptography, thus making it harder for attackers to capture the complete cryptographic keys. Following the motivation and overview, Kantarcioglu provided some background on secret sharing and threshold cryptography (e.g., Distributed-RSA, Threshold-RSA, and Shamir secret sharing).

Based on the idea of distributing the keys in many pieces, they proposed a system, called Hermes, to prevent the secret key leakage in the public cloud. As a proof-of-concept, they applied Hermes to enhance the protection of SSL/TLS cryptographic keys. In the initialization phase of this prototype, the cryptographic key is partitioned and distributed to a set of defender VMs, and Hermes is then bootstrapped with established initial authenticated and secure SSL channels between pairs of defender VMs. After that, client connection requests are sent to one of the defender VMs, named combiner VM, and the combiner VM will work with other VMs to provide services such as distributed signing and decryption. The owner of the secret key will also periodically create new shares for the same secret keys and re-share them to the VMs. These new shares are independent from the previous ones, and this re-sharing adds more difficulty for attackers to extract the cryptographic keys.

Next Kantarcioglu showed the evaluation for Hermes. Hermes was implemented as a shared library in OpenSSL, and in the experiments Hermes was used in various applications with 10 VMs in Amazon EC2. The results showed that inter-VM communication dominated the overhead, and adding defender VMs could lower the overhead. On a mail server, the overhead was at most 8% when the number of clients varied from 1 to 1000. Following the evaluation, Kantarcioglu also talked about how to formalize a multi-objective optimization problem to better choose the Hermes parameters.

Kantarcioglu was asked about the system performance compared to regular SSL, and he replied that using Hermes did introduce more overhead: for example, for one client the connection time was increased from around 2 ms to around 10 ms. He added that for bigger applications this overhead would be smaller.

Kantarcioglu was then asked about their thoughts on protecting the key-sharing process. He answered that they assume that this is a secret sharing process without any adversarial attacks.

FLUSH+RELOAD: A High Resolution, Low Noise, L3 Cache Side-Channel Attack

Yuval Yarom and Katrina Falkner, The University of Adelaide

Yuval Yarom presented a new side-channel attack against the L3 cache, called Flush+Reload, which may exist between processes in the same OS and between VMs in the cloud. Yarom started by introducing the memory-sharing mechanism, which is a popular technique for reducing the overall memory footprint and is considered safe. Yarom also provided background on the cache mechanism, especially the L3 cache, which is shared among processors, and the cache flushing functionality for maintaining cache consistency.

Yarom then talked about their Flush+Reload technique, which exploits cache behavior to infer information on victim accesses to the shared memory. In this technique, the attacker first flushes the memory line, and reloads the line after waiting for a while. The attacker then can conclude the victim memory access behavior: If the reload is short, then the victim does access the memory line during the waiting time; otherwise the victim does not access the memory line.

With the Flush+Reload technique, Yarom then showed how they attacked the GnuPG implementation of RSA. They targeted the memory lines mapped with specific code segments in the RSA program, and thus were able to trace the detailed execution of the victim program. Since the clear bits and set bits trigger different code segments in the program, the attacker can extract the secret key. Yarom showed how they traced the detailed program executions, and also showed the bit errors in their experiments on both the same-OS scenario and the cross-VM scenario. Yarom concluded with potential attack applications such as the default OpenSSL implementations of ECDSA and keystroke timing.

Someone asked how long it takes to successfully extract the key. Yarom replied under a few milliseconds. He was asked twice about whether the scheduler-based defense can defend against this attack, and his answer was no since the scheduler-based defense focuses on per-core sharing-based attacks. He was also asked about how to achieve frequent Flush+Reload. He answered that with the attacker and the victim pinned to different cores, the frequency can be high enough. He was then asked whether they can know which bits are missing in the bit errors, and why KVM missed 30 bits. Yarom replied that they can know the missing bit positions, and the high bit error rate for KVM was due to both the more advanced optimizations of the Xeon processors and the aggressive deduplication used. The last question was about whether the distributed key idea can prevent the attack. Yarom answered that having multiple collocated VMs sharing the key may be a protection for Flush+Reload attack.

Revisiting SSL/TLS Implementations: New Bleichenbacher Side Channels and Attacks

Christopher Meyer, Juraj Somorovsky, Eugen Weiss, and Jörg Schwenk, Ruhr-University Bochum; Sebastian Schinzel, Münster University of Applied Sciences; Erik Tews, Technische Universität Darmstadt

Christopher Meyer started the talk by describing the importance of SSL/TLS in security and privacy, and he stated that according to their work, oracle attacks have returned again in SSL/TLS. He then provided background about the famous Bleichenbacher attack and the handshake protocol in SSL/TLS. The Bleichenbacher attack uses the error messages of the PreMasterSecret (PMS) structure-checking algorithm as an oracle to learn whether the decrypted message of the ciphertext from the attacker started with certain bytes. After that, the attacker can leverage the RSA homomorphic property to adjust the payload iteratively and finally restore the PMS, which can be used to derive SSL/TLS session keys. This problem was addressed in the TLS 1.0 standard by using a new random PMS if there is anything wrong.

Meyer went on to report four new Bleichenbacher side channels discovered by their analysis on several widely used SSL/TLS implementations using their T.I.M.E. framework. The first side channel they discovered was due to an implementation bug in JSSE. They found that inserting 0x00 bytes at specific padding positions would generate a different error message, which can be used as the oracle. They evaluated the attack and found that 2048 bit keys were cracked using about 177k queries over a 12-hour period.

While the first side-channel exploits different error messages, the other three new side channels use different processing time as the oracle. The second side channel is likely to exist due to the countermeasure for the original Bleichenbacher attack in the OpenSSL implementation. This is because the new random PMS is generated only when there is something wrong, leading to the timing difference which leaks information about the SSL/TLS compliance of the received PMS. However, they could not execute a practical Bleichenbacher attack, because of the weakness of this oracle. The third side channel is related to the internal exception handling in JSSE. In the implementation, if the message format is not correct, an additional exception will be provoked. Although this does not trigger error messages, it can create timing difference due to the exception handling delays in Java. In the evaluation, this attack took around 19.5 hours and 18.6k queries.

The last side channel also exploits timing differences, but it exists due to hardware issues. They found that the F5 BIG-IP and IBM Datapower, which both use the Cavium NITROX SSL accelerator chip, do not verify the first byte of the message, and additionally found timing differences in processing TLS requests. Meyer showed the evaluation results, and for 2048-bit key this side channel attack took 40 hours and 7371 queries to succeed. In the end, Meyer used a table to summarize the four newly discovered side channels and their attack efficiency.

Meyer was asked whether their toolkit is made publicly available. He replied that their tool has not been released yet. He added that the timing measurement unit of the framework cannot be used ad hoc, because it has to be adjusted and tweaked for each target and environment.

Invited Talk

Summarized by Janos Szurdi (jszurdi@andrew.cmu.edu)

Battling Human Trafficking with Big Data

Rolando R. Lopez, Orphan Secure

Lopez began by discussing the models of human trafficking organizations in different countries. Lopez was an FBI agent for 15 years. During his years at FBI he gained experience about money laundering, drug trafficking, police corruption, and human trafficking. His organization focuses on four methods to battle human trafficking: Prevention and Awareness, Identification and Intervention, Restoration Care, and finally Policy and Law.

The root of human trafficking in many countries is poverty. The Chinese human trafficking model is one such example. From small and poor villages, young women and men are smuggled to big cities all around the world, with the consent of the family and with the promise of a better life. For women, most often work is offered in massage parlors, but when these women get there, their "employees" take all of their documents and they are forced into prostitution. The Chinese mafia runs underground banking and uses wire transfers, making it really hard to cut their money supplies.

In post-Soviet countries, like Russia and Ukraine, human trafficking is very brutal, where children are often sold from orphanages directly into the sex trade. Restoration of victims is really hard in these countries because these victims have no families. As opposed to the Chinese models where the human traffickers try not to harm the girls since they need to maintain a good connection with the villages, in these post-Soviet countries there are no such constraints on the violence. The only chance to help these children and women, who became victims, is for local communities to help them to regain a normal life.

In Thailand the basis of the human trafficking model is the terrible poverty, where families go hungry in certain regions because the crop wasn't enough for the entire year. This is the time when human traffickers go to these villages and offer money to the head of a family for their children, and when the fathers have to feed their other children, too, they are often willing to sell one of them to help the rest of the family stay alive. To combat human trafficking in Thailand, feeding programs can be a huge help. Feeding these children can eliminate the reason for selling them into slavery.

The most violent of all is the Albanian mafia. They lure their victims very often through alluring job offers into prostitution. They focus on women and they immediately begin drugging and raping them. They also keep law enforcement under threat, making it really hard to deal with them.

In the US model, pimps are looking for runaways, often making long-term relationships with them and helping them in the beginning, but later forcing their victims into prostitution. These victims also very often suffer from Stockholm syndrome. One newly emerging method to get new victims is the use of so-called Romeo pimps. These Romeo pimps are high school kids who are hired by real pimps. They would go to parties with these young girls and give them drugs there, and they would make pornographic videos of their victims and blackmail them with these videos into prostitution.

Human trafficking is also a very big problem in India, which is the most dangerous place to be a little girl. In India, besides children being sold for sex trafficking, they may often be mutilated to become beggars. Human trafficking in Mexico is run by international drug cartels, like the infamous MS-13, and this is just a part of their criminal portfolio, which includes money laundering and drug and gun trafficking. Finally, Nigeria and West Africa are very hard cases, where many people still believe that having sex with a young child can cure AIDS.

Lopez described a tool helpful against human trafficking called the Freedom App, where anyone experiencing anything related to human trafficking can send in information anonymously. This application already helped in freeing children held for prostitution. In addition, they have their own system that monitors all incoming information to help them with tracking and intervening in human trafficking.

An attendee asked how many false positives they get from the Freedom App. Lopez answered that they are not flooded by messages so far; in addition, before taking action, they contact local trusted officers who make sure that the information is correct. Someone asked what the community of computer security experts can do to battle human trafficking. Lopez answered that they need the most cutting-edge technology to stay ahead of criminals, but all help is welcome. Send an email with your expertise and they will help figure out how each individual can best help the cause. A questioner wondered how they determine who to trust among all the corrupt officers, especially in foreign countries. Lopez said they first contact people whom they trained in different techniques against criminals; second, they go to the local bureau to know who is trusted; most importantly, they can see whether a person is just working for the payroll or is passionate about helping victims and wants to bring justice to the criminals. For more information, go to www.orphansecure.com or write to info@orphansecure.com.

After Coffee Break Crypto

Summarized by Michael Zohner (michael.zohner@cased.de)

Burst ORAM: Minimizing ORAM Response Times for Bursty Access Patterns

Jonathan Dautrich, University of California, Riverside; Emil Stefanov, University of California, Berkeley; Elaine Shi, University of Maryland, College Park

Jonathan presented an Oblivious RAM (ORAM) system that is designed to provide quick responses under bursty workloads. When data is outsourced to the cloud, meta-information such as data access patterns can leak valuable information to the provider even when the data is encrypted. ORAM constructions were introduced to keep such access patterns hidden from malicious cloud providers. Existing ORAM constructions assume a steady stream of requests and primarily focus on minimizing the total bandwidth overhead, but real-world storage servers often have to cope with bursts of data queries. The presented Burst ORAM scheme was specifically designed to handle such bursts while minimizing response times of individual queries.

To cope with bursts of queries, Burst ORAM introduces several new techniques. First, it prioritizes the online I/O required for the client to obtain each result, delaying the more expensive shuffling I/O until idle periods between bursts. Second, it schedules shuffling jobs such that the most efficient jobs are prioritized. Third, it XORs blocks together before returning them to the client, reducing the online I/O to a constant amount.

The response times and bandwidth consumptions of Burst ORAM were simulated and compared to an insecure baseline system as well as a traditional ORAM system. The results showed that under a realistic workload with bursts of moderate lengths, Burst ORAM achieved response times that were comparable to the insecure baseline and orders of magnitude lower than traditional ORAM systems. However, Burst ORAM increases the total communication compared to traditional ORAM systems by up to 50%. Thus, a question to tackle in future work is how to reduce the overall communication overhead while keeping response times low.

Following the presentation, an audience member asked what would happen during an extremely large burst. Jonathan responded that in this case the costs of Burst ORAM would gracefully degrade toward those of traditional ORAM systems.

TRUESET: Faster Verifiable Set Computations

Ahmed E. Kosba, University of Maryland; Dimitrios Papadopoulos, Boston University; Charalampos Papamanthou, Mahmoud F. Sayed, and Elaine Shi, University of Maryland; Nikos Triandopoulos, RSA Laboratories and Boston University

Ahmed presented their work on TRUESET, a prototype for set-centric Verifiable Computation (VC) operations for outsourcing computations to a cloud. Next to privacy concerns, outsourcing work to an untrusted cloud server raises integrity and correctness concerns about computations done by the server. Verifiable computation was introduced to enable the client to verify the correctness of a computation outsourced to a remote untrusted server. Research on VC has progressed in recent years both in terms of theory and practice. However, while current schemes maintain short proofs and short verification time, the proof computation time for the server is still too high to be considered usable. This is especially the case for set-centric operations used for database queries.

TRUESET was designed to reduce the proof computation time for the server and achieve an input-specific runtime while retaining the expressiveness of previous techniques. TRUESET achieves this by representing its operations using polynomials instead of arithmetic or Boolean circuits as done by traditional approaches. The polynomial circuit encodes the input size as a degree of the polynomial. Thereby, the circuit size becomes constant and independent of the size of the sets. Furthermore, special transformation gates can be used to transform the polynomial representation of sets into an arithmetic representation if additional non-set-centric operations are required.

TRUESET was implemented and compared to current systems for VC. Most prominently, TRUESET achieved more than 30x speed-up for the proof computation runtime compared to existing approaches for sets with more than 64 elements, reaching 150x speed-up for a union of two 256-element sets. Furthermore, TRUESET was shown to be applicable to sets with 30x larger size than previous approaches. Despite the large speed-up, Ahmed pointed out that their implementation was not yet practical, but is meant to spawn interest in practical VC systems.

Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture

Eli Ben-Sasson, Technion—Israel Institute of Technology; Alessandro Chiesa, Massachusetts Institute of Technology; Eran Tromer, Tel Aviv University; Madars Virza, Massachusetts Institute of Technology

Madars presented work on generating non-interactive zero-knowledge proofs that are short and easy to verify and rely on a setup that is independent of the proven function. When computing on decentralized information, we often encounter the problem that the goals of integrity and confidentiality are complementary to each other. For instance, if a server holds a confidential database and a client wants to evaluate a public function on his public input and the database, the client either has to trust the server to compute the correct output or the server has to disclose his database to the client. To enable this computation while bridging the gap between integrity and confidentiality, zero-knowledge proofs can be used. A zero-knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK) is a special type of zero-knowledge system that builds short and easy-to-verify non-interactive zero-knowledge proofs that are based on suitable cryptographic assumptions. While practically feasible zk-SNARKs for certain applications exist, they rely on a costly trusted setup that is tailored to the evaluated function. Furthermore, they have only limited support for high-level languages.

Instead of being targeted to a particular function, Madars introduced a zk-SNARK system with universal setup that uses universal circuits to compute a proof for the desired functionality. The system takes as input bounds on the program size, the input size, and the time, and the corresponding trusted setup allows supporting all program computations that are within these bounds. However, in prior work, generating a universal circuit incurs a multiplicative overhead in the program size and is therefore only feasible for small programs. To allow the evaluation of larger programs, a routing network was introduced, which reduced the multiplicative to an (essentially) additive overhead.

The above circuit generator is composed with a zk-SNARK system for circuits. Both components were implemented and their performance was compared to existing approaches. As for the zk-SNARK for circuits, when evaluated on a one million-gate circuit with a thousand-bit input, its proof generation time was shown to be 5.3x faster and its proof verification time was shown to be 1.8x faster. The implementation was used in Zerocash, a privacy-preserving digital currency. Madars and his coauthors are looking for further applications of their work.

An audience member asked whether the authors had looked at universal circuit generators in the literature instead of constructing a new one. Madars replied that one could consider such computational models, but as their goal was SNARKs for RAM computations, it would incur a sub-optimal intermediate reduction: from RAM to circuits, followed by a universal circuit for circuits; they chose to have universal circuits that directly support RAM computations.

Faster Private Set Intersection Based on OT Extension

Benny Pinkas, Bar-Ilan University; Thomas Schneider and Michael Zohner, Technische Universität Darmstadt

Michael surveyed and optimized existing protocols for Private Set Intersection (PSI) in the semi-honest model and presented a novel protocol based on Oblivious Transfer (OT). PSI considers the problem of two parties each holding a set of elements and wanting to identify the elements they have in common without disclosing any other element. PSI protocols can, for instance, be used for secure database joins or discovery of common contacts. Since PSI is a general problem and is often used as an indicator for the practicality of secure computation, it has gained a lot of attention, and many protocols based on different techniques have been proposed. However, there have been various inconsistencies in the evaluation and comparison of the protocols, resulting in uncertainty about the best performing protocol for a particular deployment scenario.

Major results on PSI protocols were outlined and categorized depending on their underlying technique as public-key-based, generic secure-computation, or circuit-based and OT-based. Using current state-of-the-art techniques in secure computation, optimizations for a circuit-based and an OT-based protocol were proposed, which decreased both their runtime and communication by at least a factor of 2. Michael then introduced a new

OT-based PSI protocol that made use of recent improvements for OT extension and used hashing schemes to achieve better efficiency.

Finally, the performance of all surveyed protocols was evaluated using the same programming language, techniques, libraries, and benchmarking environment. The results showed that the public-key-based protocols had a moderate runtime for long-term security but had the most efficient communication complexity. The circuit-based protocols had the highest runtime and communication complexity but could be extended to other functionalities without requiring a proof-of-security. The OT-based protocols achieved the best overall runtime. To evaluate the practical usability of PSI, the results were compared to the performance of the naive hashing solution that is currently used in practice but that leaks information about the inputs. Compared to the best performing PSI protocol, the naive solution had an order of magnitude less runtime and communication.

The first question concerned the possibility of authenticating the input elements that are used by each party. Michael replied that although his work did not ensure authenticity of input elements, existing approaches could be used to achieve this property. The second questioner wanted to know how the results of the paper "Private Set Intersection: Are Garbled Circuits Better than Custom Protocols?" in NDSS '12 fit in with the overall results of the presented work. Michael replied that the results of the NDSS '12 paper were later revised in the paper "Experimenting with Fast Private Set Intersection" in TRUST '12, leading to confusion about the performance of the analyzed schemes.

Program Analysis: Attack of the Codes

Summarized by Brendan Saltaformaggio (bdsaltaformaggio@gmail.com)

Dynamic Hooks: Hiding Control Flow Changes within Non-Control Data

Sebastian Vogl, Technische Universität München; Robert Gawlik and Behrad Garmany, Ruhr-University Bochum; Thomas Kittel, Jonas Pfoh, and Claudia Eckert, Technische Universität München; Thorsten Holz, Ruhr-University Bochum

Sebastian Vogl presented a new approach, called dynamic hooks, to construct malicious code hooks residing purely in non-control data. As a running example, the Linux kernel's list_del function was used to illustrate a dynamic hook. By crafting a special list node to be deleted, an attacker could cause an overwrite of a kernel return address or other control flow data. Using a combination of program slicing and symbolic execution, the researchers revealed 566 and 379 execution paths in the Linux and Windows kernels, respectively, that are vulnerable to such attacks.

The authors' choice to use VEX was questioned. Vogl responded that they had tested others, but VEX worked best without any practical problems. William Enck, the session chair, asked how common are side effects in different paths. Vogl admitted that it requires an expert to look at a specific exploit path to determine whether that specific path works for a given attack.

X-Force: Force-Executing Binary Programs for Security Applications

Fei Peng, Zhui Deng, Xiangyu Zhang, and Dongyan Xu, Purdue University; Zhiqiang Lin, The University of Texas at Dallas; Zhendong Su, University of California, Davis

Fei Peng began his presentation citing the limitations of current binary analysis frameworks: Static analysis suffers from over-approximation and lack of runtime data. Dynamic analysis lacks coverage. Symbolic execution may not scale. Given these limitations, X-Force attempts to force a binary to execute as many code paths as possible by dynamically flipping select conditional branches.

Peng argued that, while X-Force is not complete or sound, it overcomes the limitations of current binary analysis platforms. X-Force is therefore designed as a new platform that binary analysis tools can be built upon. The presentation showed how X-Force could achieve far better CFG construction and indirect call coverage on most of the SPEC test suite binaries, compared to traditional static or dynamic analysis. Finally, Peng showed how an existing type reverse engineering tool (REWARDS) was ported to X-Force, and this caused a considerable increase in variable coverage.

The first questioner commented that X-Force is similar to concolic execution systems except that X-Force chooses to execute invalid paths to avoid the slowdown of concolic execution, and asked whether they had compared X-Force with any concolic execution systems. Peng replied that they did compare X-Force with the S2E system and that the results are in the paper. Peng was then asked whether X-Force can utilize parallelization speed-up runs for online use. His response was that X-Force already can parallelize multiple runs.

ByteWeight: Learning to Recognize Functions in Binary Code

Tiffany Bao, Jonathan Burket, and Maverick Woo, Carnegie Mellon University; Rafael Turner, University of Chicago; David Brumley, Carnegie Mellon University

Given the importance and difficulty of automatically identifying functions within binaries, Tiffany Bao presented a new solution called ByteWeight. For motivation, Bao showed how different levels of compiler optimizations break many common function boundary signatures. Further, she explained that the static binary analysis tool IDA is often unable to uncover functions at such high levels of optimization.

To address the function identification challenge, Bao described how ByteWeight combines machine learning and program analysis to perform function identification. First, training binaries are used to extract common function prefix sequences and build a weighted prefix tree. These weighted prefix trees are then used to identify function boundaries within test binaries.

Bao then presented the results of applying ByteWeight to 2200 binaries. Comparing against a previous approach by Rosenblum et al., ByteWeight uncovered far more functions within the test binaries. When compared to other function start identification tools, ByteWeight's precision and recall was again higher. Lastly, Bao invited others to test the ByteWeight system by downloading a preconfigured VM at http://security.ece.cmu.edu/byteweight/.

Bao was asked why they chose to use the BAP platform over an IR. She responded that BAP was chosen because much of their implementation is based on BAP and that ByteWeight could also be ported to use other platforms. Eric Eide (University of Utah) noted that the paper's experiments are not compiler-specific and asked whether the results would be better if ByteWeight was trained on a specific compiler. Bao replied that the difference may not be large and that they do not need compiler-specific knowledge. Finally, a questioner asked how many prologues were in a common weighted prologue tree. Bao responded that in their current results, trees often contain thousands of nodes.

Optimizing Seed Selection for Fuzzing

Alexandre Rebert, Carnegie Mellon University and ForAllSecure; Sang Kil Cha and Thanassis Avgerinos, Carnegie Mellon University; Jonathan Foote and David Warren, Software Engineering Institute CERT; Gustavo Grieco, Centro Internacional Franco Argentino de Ciencias de la Información y de Sistemas (CIFASIS) and Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET); David Brumley, Carnegie Mellon University

With the popularity of fuzzing as a software testing technique, Sang Kil Cha presented a new effort to mathematically reason about how to pick the best seeds for fuzzing. "Best" here is finding the most bugs with specific fuzzing seeds. The presentation then explained the several different selection algorithms developed in the paper and how each of these algorithms compared to one another. Finally, the results of fuzzing a variety of applications with different seed-selection algorithms is shown, and the effectiveness of each algorithm is compared.

The session chair, William Enck, noted that the selection of seeds assumes that you have a base data set and asked how it handles applications that modify the fuzzed data. Sang Kil Cha responded that the current framework already handles such scenarios and that this does not affect their results.

After Lunch Break Crypto

Summarized by Michael Zohner (michael.zohner@cased.de)

LibFTE: A Toolkit for Constructing Practical, Format-Abiding Encryption Schemes

Daniel Luchaup, University of Wisconsin—Madison; Kevin P. Dyer, Portland State University; Somesh Jha and Thomas Ristenpart, University of Wisconsin—Madison; Thomas Shrimpton, Portland State University

Kevin presented LibFTE, an easy-to-use toolkit for instantiating Format-Preserving Encryption (FPE) and Format-Transforming Encryption (FTE) schemes. Several applications, such as in-place encryption of credit card numbers in a database, require that plaintexts and ciphertexts abide by a specific format. A related problem, which was investigated by the authors in their previous work "Protocol Misidentification Made Easy with Format-Transforming Encryption" at CCS '13, focuses on circumventing network censors by using FTE to transform ciphertexts into messages that are indistinguishable from real network protocol messages. Kevin revisited the notion of FTE and outlined how to instantiate an FTE scheme.

While FPE/FTE schemes exist, instantiating them for a particular format requires expert knowledge as well as engineering expertise to achieve good performance. LibFTE was designed to reduce the challenges in building FPE/FTE schemes. LibFTE allows the developer to specify the format of the plaintext and the format of the ciphertext separately using regular expressions. However, using prior techniques, to instantiate a FPE/FTE scheme, the regular expressions have to be transformed to a deterministic finite automaton (DFA), which can grow exponentially in size and can even require around 200 MB for some formats. To reduce the memory requirements for instantiating FPE/FTE schemes, the concept of relaxed ranking was introduced. In relaxed ranking, FPE/FTE encryption is performed directly from the nondeterministic finite-state automaton (NFA) representation of the regular language, which obviates the need for NFA to DFA conversion.

LibFTE has interfaces for C++, Python, and JavaScript and comes with a configuration assistant for instantiating FPE/FTE schemes. LibFTE was evaluated in several scenarios. Firstly, it was tested on 3,500 regular expressions from the Snort IDS, where it was able to instantiate FPE/FTE schemes for all expressions using less than 150 MB memory, even when FPE/FTE schemes without relaxed ranking failed. In addition, the average required memory was reduced by 30%. Secondly, its performance for simultaneous compression+encryption was compared to regular AES encryption using PostgreSQL, where it saved 35% disk space on average while maintaining a similar query time. Thirdly, a LibFTE Firefox extension was implemented and used to encrypt a Yahoo! address book contact form. LibFTE is available online at https://libfte.org.

Ad-Hoc Secure Two-Party Computation on Mobile Devices using Hardware Tokens

Daniel Demmler, Thomas Schneider, and Michael Zohner, Technische Universität Darmstadt

Daniel presented his work on practical secure computation on mobile devices using a smart card to achieve more efficiency. Mobile devices have become an important tool in modern society. They measure, collect, and store various data throughout the daily life of their user and are used to perform various tasks. This makes them an important target for privacy measures such as secure computation. Secure computation enables two parties to evaluate a public function on their private inputs without leaking any information about either party's input except what can be obtained from the result. While secure computation on desktop PCs is becoming increasingly practical, its runtime on resource-constraint mobile devices is still too high to be considered usable.

To increase performance of secure computation on mobile devices, Daniel presented a scheme that uses a smart card, located on one device, as trusted third party. The scheme is secure against passive adversaries, works in three phases, and offloads the bulk of the workload of the secure computation protocol to the smart card. In the first phase, called the init phase, the smart card pre-generates the data required for the secure computation. The init phase can be performed independently by the device holding the smart card at any point in time: for instance, when the device is being charged. In the second phase, called the setup phase, the smart card securely transmits the helper data, generated in the init phase, over a secure channel to the partner device the secure computation protocol is executed with. In the third phase, called the online phase, the devices run the secure computation protocol and obtain the output.

The presented scheme was implemented, evaluated, and compared to related work on three example applications using an Android smartphone and an off-the-shelf smart card. In the first application, which considered privacy-preserving scheduling of a meeting, the smart-card-based scheme achieved 3x better runtime than existing non-smart-card-based schemes. In the second application, the scheduling functionality was extended by location-awareness, demonstrating that the computed function could easily be changed. The third application considered private set-intersection and showed that the smart-card-based scheme achieved a similar performance on smartphones as a non-smart-card aided secure computation protocol execution on desktop PCs. The evaluation demonstrated that secure computation on resource constrained mobile devices can indeed be practical when supported by a trusted hardware token.

Someone asked Daniel whether an extension of his scheme to stronger adversaries was possible. Daniel replied that an extension to malicious adversaries would be possible by also equipping the partner device with a hardware token and massaging the TinyOT protocol introduced in “A New Approach to Practical Active-Secure Two-Party Computation” at CRYPTO ’12.

ZØ: An Optimizing Distributing Zero-Knowledge Compiler

Matthew Fredrikson, University of Wisconsin—Madison; Benjamin Livshits, Microsoft Research

Matthew presented work on ZØ, a zero-knowledge protocol compiler that combines two existing zero-knowledge systems and translates from C# to distributed multi-tier code. To maintain a client’s privacy, several applications move functionality to the client and send only aggregated outputs to a server. However, the client can not necessarily be trusted to return the correct output to the server. Hence, there is a tradeoff between the privacy of the client and the integrity of the output. Google Waze was mentioned as a prominent example for this tradeoff, where users provide their traffic data to improve quality of routing information.

Zero-knowledge protocols are a common tool to fulfill the two complementary goals of privacy and integrity. To ease the development of zero-knowledge protocols, zero-knowledge protocol compilers, such as Pinocchio and ZQL, have been introduced. However, these compilers are directly based on one particular technique for generating zero-knowledge proofs and scale very poorly to large applications and to applications that cannot efficiently be expressed using the underlying technique.

The main focus of ZØ is to make the generation of efficient zero-knowledge protocols easier. ZØ compiles from C# and allows developers to specify zero-knowledge regions using LINQ syntax. To support code-generation for different distributed scenarios, ZØ allows splitting functionality among multiple tiers. Additionally, it generates more efficient zero-knowledge protocols that scale to larger problems by not tying itself to a particular zero-knowledge technique. This is done by combining both the techniques that are used in Pinocchio and ZQL and using a cost model to schedule the techniques across multiple tiers such that the resulting protocol obtains the most efficient runtime.

To evaluate the performance benefits of ZØ, six different real-world applications were implemented in ZØ, Pinocchio, and ZQL, and the performance of the resulting protocols was compared. Overall, ZØ allowed to scale up to 10x larger application sizes and resulted in up to 40x faster runtime compared to Pinocchio and ZQL.

SDDR: Light-Weight, Secure Mobile Encounters

Matthew Lentz, University of Maryland; Viktor Erdélyi and Paarijaat Aditya, Max Planck Institute for Software Systems (MPI-SWS); Elaine Shi, University of Maryland; Peter Druschel, Max Planck Institute for Software Systems (MPI-SWS); Bobby Bhattacharjee, University of Maryland

Matthew outlined the Secure Device Discovery and Recognition (SDDR) system for short-range secure mobile encounters. Many mobile social services, such as Haggle and Foursquare, detect nearby peers (strangers and/or friends) and support communication among these peers. Traditional approaches that solve this problem either use a centralized service or a decentralized approach relying on device-to-device communication with static addresses (or IDs), which both allow tracking users’ movements. Using random IDs, on the other hand, prevents the device from being recognizable by friendly devices.

SDDR avoids tracking while allowing the recognition of friendly devices by using a decentralized approach in combination with random user IDs and cryptographic techniques. Existing cryptographic techniques, however, perform poorly on resource-restricted devices such as mobile phones, both in terms of runtime and energy consumption. Matthew outlined a novel non-interactive solution that uses the Bluetooth controller to send a beacon message upon query. This beacon enables secure communication between peers and can allow permitted friends to recognize the device. In addition to granting people the right to recognize a device, SDDR also introduces methods to efficiently revoke this permission.

SDDR was implemented on an Android device, and its runtime was measured as well as its energy consumption. The runtime for a single query amounted to less than one millisecond, four orders of magnitude faster than the protocol that uses existing cryptographic techniques. The energy consumption was measured over a period of 30 minutes and extrapolated to the total consumption for a day. While SDDR consumed around 10% of the phone’s battery capacity, the protocol based on existing cryptographic techniques required 191% (depleting the battery in half a day). As an example for further applications, such as facilitating communication among groups of peers who encountered each other presently (or in the past), a reference to the authors' paper "EnCore: Private, Context-based Communication for Mobile Social Apps" was given. The code for SDDR is available online at http://www.cs.umd.edu/projects/ebn/.

Program Analysis: A New Hope

Summarized by Lucas Davi (lucas.davi@trust.cased.de)

Enforcing Forward-Edge Control-Flow Integrity in GCC & LLVM

Caroline Tice, Tom Roeder, and Peter Collingbourne, Google, Inc.; Stephen Checkoway, Johns Hopkins University; Úlfar Erlingsson, Luis Lozbno, and Geoff Pike, Google, Inc.

Caroline Tice presented a compiler-based Control-Flow Integrity (CFI) approach to prevent runtime attacks. Caroline argued that return addresses and stack data are today well-protected due to stack canaries and address space layout randomization. Hence, attackers typically corrupt vtable or function pointers to launch a runtime attack. To tackle these attacks, Caroline presented a compiler-based approach that enforces so-called forward-edge control-flow integrity (CFI) which instruments indirect calls and jumps with CFI checks. Specifically, she presented VTV (Virtual-Table Verification) for GCC, and IFCC (Indirect Function-Call Checks) for LLVM. The presented CFI solution is open source, induces a modest overhead of 1–8.7%, and protects 95–99.8% of all indirect function calls.

Lucas Davi (TU Darmstadt) asked whether forward-edge CFI prevents an adversary from exploiting an indirect call or jump to invoke VirtualAlloc or VirtualProtect in a modern application like Adobe Reader. Caroline responded that this depends on the virtual calls used by the target application. David Evans (University of Virginia) asked at which time forward-edge CFI will be applied to the Chrome browser. Caroline mentioned that this is the goal of the project, but there are still some issues to tackle.

ret2dir: Rethinking Kernel Isolation

Vasileios P. Kemerlis, Michalis Polychronakis, and Angelos D. Keromytis, Columbia University

Vasileios started with an introduction to kernel exploits and mentioned that most existing kernel attacks are based on exploiting a memory corruption vulnerability in the kernel to redirect execution to a shellcode or ROP payload residing in user space. Such attacks are referred to as ret2usr attacks and can be detected by new hardware and compiler-based defenses (SMEP, SMAP, PXN, KERNEXEC, UDEREF, and kGuard) that basically all prevent the kernel from either executing code or tampering with reference data from userland. However, Vasileios demonstrates that all these defenses can be bypassed with a new attack technique called return-to-direct-mapped memory (ret2dir). The main idea is to redirect execution to a kernel memory region called physmap that contains a direct mapping of all the physical memory in the system (including pages mapped from user space). Hence, the attacker only needs to know where his shellcode is mapped to in the physmap region to execute the shellcode from the kernel space. To defend against ret2dir attacks, Vasileios presented exclusive page frame ownership (XPFO) that unmaps userland pages from physmap and remaps them (deleting the page contents) when they are reclaimed by user space.

David Evans (University of Virginia) asked whether sharing of data between kernel and user space is common in modern systems as this would bypass the presented defense against ret2dir attacks. Vasileios replied that it depends on how frequently page caching is used by the system. Someone asked whether existing kernel data integrity protection mechanisms would prevent ret2dir attacks. Vasileios confirmed that such integrity mechanisms would also defend ret2dir attacks. Someone asked whether these attacks were also possible in Windows. Vasileios replied that Windows also has a page cache and the attacks are not specific to Linux.

JIGSAW: Protecting Resource Access by Inferring Programmer Expectations

Hayawardh Vijayakumar and Xinyang Ge, The Pennsylvania State University; Mathias Payer, University of California, Berkeley; Trent Jaeger, The Pennsylvania State University

Hayawardh started with a motivating example to demonstrate that resource access control is an important problem. In particular, he showed how a university's Apache Web server can be compromised by a student (who owns a page on the Web server) to retrieve the password file exploiting a symbolic link to the password file. Hayawardh elaborated on two reasons why such attacks were still possible: (1) Programmers, administrators, and OS distributors have different expectations on how files are protected and used on the system, and (2) code complexity makes it challenging for a programmer to protect every resource used in his program. To tackle this security problem, Hayawardh presented a solution to map programmer expectation onto a system. In particular, Hayawardh presented a process firewall based on introspection of the program limits system calls to their appropriate resources as given by their original intent and expectation. As a motivation to deploy the process firewall, the authors' evaluation showed that for four of five tested programs, more than 55% of the resource accesses were not protected with defensive checks or filters.

Rob Johnson (Stony Brook) asked Hayawardh how policies were generated and about the advantages of using the process firewall compared to fixing the code. Hayawardh replied that the process firewall runs automated methods to generate the corresponding access control policies.

Static Detection of Second-Order Vulnerabilities in Web Applications

Johannes Dahse and Thorsten Holz, Ruhr-University Bochum

Facebook Internet Defense award!

Johannes presented a static analysis tool that detects second-order vulnerabilities in Web applications. He started by describing first-order vulnerabilities such as the well-known SQL injection attack. These attacks can be prevented by applying sanitization. However, second-order vulnerabilities occur when an attack payload is first persistently stored in a database or file, and the application reads in a second stage the payload to perform a security-critical operation. To identify these vulnerabilities, Johannes presented a static source code analysis tool (focusing on PHP) that analyzes write and read operations of an application to persistent data. In general, the static analysis tool builds a control-flow graph of the application, identifies sensitive sinks, and validates whether user input can potentially write to a sensitive sink recording and considering also possible sanitization on data inputs. The same is also done for inputs that originate from persistent data storage to cover read operations by the application. Finally, data input writes and reads are correlated with each other to connect input and output points to identify second-order vulnerabilities. The evaluation of six Web applications showed that the static analysis approach identified 159 true positives (second-order vulnerabilities) and 43 false positives.

Benjamin Livshits (MSR) asked Johannes about the lessons learned, and why the analysis did not take other languages beyond PHP into account. Johannes replied that in contrast to Java or other languages, PHP is particularly vulnerable to second-order vulnerabilities. However, it was still possible to write PHP-secure code. Someone mentioned that static analysis has limitations because it misses dynamic behavior. Johannes noted that there were some false positives and that the presented approach focuses on vulnerabilities that can be detected at static analysis time.

Mobile Apps and Smart Phones

Summarized by Shouling Ji (sji@gatech.edu)

ASM: A Programmable Interface for Extending Android Security

Stephan Heuser, Intel CRI-SC at Technische Universität Darmstadt; Adwait Nadkarni and William Enck, North Carolina State University; Ahmad-Reza Sadeghi, Technische Universität Darmstadt and Center for Advanced Security Research Darmstadt (CASED)

Stephan Heuser presented work on promoting OS security extensibility in the Android OS. Specifically, the authors designed a framework named Android Security Modules (ASM) that provides a programmable interface for defining new reference monitors for Android. In the presentation, Heuser first demonstrated the architecture of Android, which consists of three layers (from top to bottom): the App layer, the Android OS layer, and the Linux kernel layer. Currently, to improve the security of the Android system, access control is implemented in every layer. Subsequently, to motivate their work, Heuser summarized over a dozen recent Android security architecture proposals to identify the hook semantics requirements for Android security models. Based on their survey, they concluded that Android OS is responsible for enforcing more than just UNIX system calls. They also identified that it is necessary for authorization hooks to replace data values and for third-party applications to introduce new authorization hooks.

Based on their findings, they designed and implemented an extensible Android Security Modules (ASM) framework. Heuser introduced how ASM works layer by layer. Basically, ASM allows multiple simultaneous ASM apps to enforce security requirements while minimizing the system overhead. To demonstrate the utility and performance of the proposed ASM framework, they implemented several ASM apps. In the presentation, Heuser showed one ASM app MockDroid, which is a system-centric security extension for the Android OS, allowing users to gracefully revoke the privileges requested by an application without the app caching. Besides that, Heuser also demonstrated the performance overhead and energy consumption of the ASM framework.

Following Heuser's talk, there was an interesting discussion on the ASM framework. First, someone was curious about the overall design and the novelty of ASM, especially the work mechanism of ASM in the kernel layer. Heuser summarized the design of ASM and highlighted the distinguished features of ASM. He also directed the audience to find more discussion in the paper. Another attendee was curious about how ASM implements the isolation of apps. Based on Heuser's response, ASM does not separate apps (or components) in its current version. It is an interesting future research direction of this paper.

Brahmastra: Driving Apps to Test the Security of Third-Party Components

Ravi Bhoraskar, Microsoft Research and University of Washington; Seungyeop Han, University of Washington; Jinseong Jeon, University of Maryland, College Park; Tanzirul Azim, University of California, Riverside; Shuo Chen, Jaeyeon Jung, Suman Nath, and Rui Wang, Microsoft Research; David Wetherall, University of Washington

Jaeyeon Jung presented their solution to the problem of third-party component integration testing at scale, in which one party wishes to test a large number of applications using the same third-party component for a potential vulnerability. First, Jung analyzed the status quo of the use of third-party components and why they are commonly used. Subsequently, Jung pointed out that the use of third-party components may cause some security risks, which have been demonstrated in several existing reports. Aiming at understanding the security of third-party components, they designed an app automation tool named Brahmastra for helping app stores and security researchers test third-party components in mobile apps at runtime.

Jung motivated their design by analyzing the limitations of existing third-party component testing tools. Taking Monkey as an example, Jung showed its vulnerability step by step using a demo in which Monkey leaked people's Facebook profiles. Then Jung presented their approach, which leverages the structure of Android apps to improve test hit rate and execution speed. The core techniques of their approach include two aspects: They characterize an app by statically building a page transition graph and call chains, and they rewrite the app under test to directly invoke the callback functions that trigger the desired page transitions.

Jung also showed the design and implementation of their testing tool Brahmastra. To evaluate the performance of Brahmastra, they tested 1010 popular apps crawled from Play Store. According to their results, Brahmastra significantly outperforms the existing solution PUMA with respect to the hit rate and test speed. Finally, Jung also demonstrated their security analysis in two scenarios: ads in kids' apps and social media add-ons. Based on their testing results, 175 out of 220 children's apps point to Web pages that attempt to collect personal information, which is a potential violation of the Children's Online Privacy Protection Act (COPPA); and 13 of the 200 apps with the Facebook SDK are vulnerable to a known access token attack.

Someone pointed out that Brahmastra is a goal-driven tool. It might be unfair to compare it with PUMA. Jung responded that Brahmastra is designed to help app stores and security researchers test third-party components in mobile apps at runtime. Therefore, both Brahmastra and PUMA have their advantages and disadvantages. Another person asked about the code crush of Brahmastra. Jung pointed out that if the app resists being rewritten, it is possible that the program is crushed.

Peeking into Your App without Actually Seeing It: UI State Inference and Novel Android Attacks

Qi Alfred Chen, University of Michigan; Zhiyun Qian, NEC Laboratories America; Z. Morley Mao, University of Michigan

Qi Alfred Chen explained that on the Android system, a weaker form of GUI confidentiality can be breached in the form of UI state by a background app without requiring any permissions. First, Chen explained the importance of GUI security. Since GUI content confidentiality and integrity are critical for end-to-end security, the security of smartphone GUI frameworks remains an important topic. Then, Chen showed a weaker form of GUI confidentiality breach, which might enable UI state hijacking. Through a demo, Chen demonstrated how UI state hijacking attack steals people's passwords in the H&R Block app. In addition, Chen also showed that this can enable other attacks, which can be classified as UI state inference attacks.

Chen summarized the underlying causes of such UI state inference attacks. This is mainly because the Android GUI framework design leaks UI state changes through a publicly accessible side channel, which is a newly discovered shared-memory side channel. Based on this finding, Chen showed the general steps of UI state inference attacks, which mainly consist of three steps: activity transition detection; activity inference; and UI state hijacking, camera peeking, and other UI state inference attacks. To detect the activity transition, the newly discovered shared-memory side channel will be employed. For activity inference, besides the newly discovered shared-memory side channel, other side channels, e.g., CPU, network activity, will also be used. Finally, they also evaluated the UI state inference attacks on seven popular Android apps, including WebMD, Chase, Amazon, Newegg, Gmail, and H&R Block. The results show that for six of the seven apps, the UI state inference accuracies are 80–90% for the first candidate UI states, and over 93% for the top three candidates.

Following Chen's talk, several concerns were raised. First, someone was curious about whether the attacks were reported to Google. Chen responded that the system is still under testing. They will report the vulnerability to Google after finishing all the tests. Another attendee asked whether the presented attacks were device-specific. Chen said they tested the attacks on popular devices and it should be easier to implement the attacks on new devices.

Gyrophone: Recognizing Speech from Gyroscope Signals

Yan Michalevsky and Dan Boneh, Stanford University; Gabi Nakibly, National Research & Simulation Center, Rafael Ltd.

Yan Michalevsky presented a new attack for recognizing speech from gyroscope signals generated from iOS and Android phones. In the talk, Michalevsky first introduced how a MEMS gyroscope works and presented the initial investigation of its properties as a microphone. Subsequently, Michalevsky showed that the acoustic signal is sufficient to extract information of the speech signal, e.g., speaker characteristics and identity. The extraction leverages the fact that aliasing causes information leaks from higher frequency bands into the sub-Nyquist range. Third, Michalevsky demonstrated that isolated word recognition can be improved if the gyroscopes of multiple devices that are in close proximity can be sampled. They also evaluated their approach by repeating the speaker-dependent word recognition experiment on signals reconstructed from readings of two Nexus 4 devices. Finally, several suggestions were made to mitigate the potential risks of such an attack.

A questioner wondered about the countermeasures of limiting the sensors/meters of smartphones. Michalevsky summarized present solutions and proposed possible future research directions. An attendee pointed out that some designs have been proposed recently to prevent apps from reading (critical) sensor readings. Michalevsky responded that even so, there are still a lot of apps that have such specific permissions to access sensor readings. Therefore, such attacks are possible and very likely to happen in the real world. Finally, a questioner wondered whether the authors considered other sensor/meter readings. Michalevsky confirmed that they also looked at other sensor/meter readings. In this paper, however, they focused on recognizing speech from gyroscope signals.

Panel

The Future of Crypto: Getting from Here to Guarantees

The summary of this session is available online as an electronic supplement: www.usenix.org/login/dec14.

The complete summaries from CSET '14, 3GSE '14, FOCI '14, HealthTech '14, and WOOT '14 are available online as electronic supplements: www.usenix.org/login/dec14.