;login:
DECEMBER 2013 VOL. 38, NO. 6
ELECTRONIC SUPPLEMENT

Conference Reports

Sec '13: 22nd USENIX Security Symposium

Washington, D.C.
August 14–16, 2013

Summarized by: Theodore Book, Sven Bugiel, Stephen Crane, Rik Farrow, Xinyang Ge, Frank Imeson, Bhushan Jain, Muhammad Naveed, Rahul Pandita, John Scire, Matthias Wählisch, Gang Wang, Yuchen Zhou, Ziming Zhao, and Bin Zeng

Opening

Summarized by Rik Farrow (rik@usenix.org)

Sam King, the Program Chair, told us that there were 277 submissions to the 22nd Security Symposium, and that 44 had been accepted. After thanking the program committee members, Sam suggested that we not miss the rump session on Wednesday night (which turned out to be both a lot of fun and interesting).

The Best Paper award went to "Control Flow Integrity for COTS Binaries," by Mingwei Zhang and R. Sekar (Stony Brook University). The Best Student Paper award was presented to "Securing Computer Hardware Using 3D Integrated Circuit (IC) Technology and Split Manufacturing for Obfuscation," by Frank Imeson, Ariq Emtenan, Siddharth Garg, and Mahesh V. Tripunitara (University of Waterloo). Finally, Sam presented Crispin Cowen with the Test of Time award for Stackguard, a mechanism that guards against stack overflows and that Crispin led the development of over ten years ago.

Wednesday Keynote Address

Summarized by Rik Farrow (rik@usenix.org)

Dr. Felten Goes To Washington: Lessons from 18 Months in Government

Edward W. Felten, Director, Center for Information Technology Policy, and Professor of Computer Science and Public Affairs, Princeton University; former Chief Technologist, U.S. Federal Trade Commission

Ed Felten worked a year and a half as the Chief Technologist at the Federal Trade Commission (FTC). He explained what it was like to work with buildings full of lawyers, and what we can do to work more effectively with these people. There are differences in culture, and he adopted the mode of dress of Washington people. Emphasizing this point, Felten removed his suit coat, tie, and dress shirt and revealed a tee-shirt showing Evi Nemeth's nine protocol layers, a more appropriate style of dress for USENIX conferences than the coat-and-tie of Washington.

Felten pointed out that Senator Ted Stevens got into trouble for describing the Internet as a series of tubes. But this was not that ridiculous, as we had talked about networks as pipes all the time. We still believe that politicians don't get it, but then they stereotype us as well. Felten displayed a picture of a kid in his parent's basement with cigarette smoke-stained PC XT. People in Washington did notice the SOPA and PIPA protests, so the people here do believe they need to pay attention to us. But this is still an awkward problem, how to meet and work with us. Felten used a photo of Elvis shaking hands with President Nixon as an example (https://en.wikipedia.org/wiki/File:Elvis-nixon.jpg); we need to be like Elvis and learn how to work with Nixon.

Felten then explained his job. The FTC missions are consumer protection and antitrust/competition (shared with DoJ) and involve civil law enforcement and investigation. Felten acted as the policy advisor to the Chairman of the FTC, as an internal technology consultant in the agency, and finally, as an ambassador to the tech community.

At this point, Felten got really interesting as he explained politics using examples from set theory and algorithms. In our culture, we are obliged to pretend to agree on truth and to learn from each other, instead of using overheated rhetoric and bogus claims. But politics is not a search for truth and this is a feature rather than a bug. Democracy is not a search for truth but an algorithm for resolving disagreements—voting. With voting, all questions are decidable in constant time (O(1)). There is no need to decide issues based on underlying facts or coherent explanations.

Individual legislators appear to be logically inconsistent and indifferent to truth, but politicians behave that way for a reason. Felten then showed pseudocode to explain how politicians can appear inconsistent. He proposed that voters have a "utility function" that allows them to like or dislike bills, and he made assumptions: that the behavior of voters is sensible, and that their ratings on two disjoint bills is disjoint. Felten went on to show that because voters can like, or dislike, bills by differing amounts, it is possible for a combination of two disjoint bills to fail passage because the degree of dislike for one part of the bill is greater than the degree of "like" for the remainder of the bill. The result is that the outputs of democracy are not logically consistent. Felten expounded on this model, showing that if legislatures follow majority opinion, they will also be logically inconsistent and appear indifferent to the truth—because they are. He also pointed out that the problem of adding amendments to bills is NP complete.

Policy-makers need to be generalists, as they have a very broad domain to cover and they can't be an expert in every area. Their goal is to make good decisions, and to do so they need to be clueful. Felten presented his ladder of cluefulness. The bottom rung is to recognize that expertise exists. The middle step is to recognize true experts, and the top step involves working effectively with experts. The top rung for experts is to work effectively with decision-makers. To do so, you learn about their knowledge and preferences and provide information to help fill in the structure of the decision space.

If you have reached the right point in your career, consider taking a sabbatical and working for the government, just as Felten and Steve Bellovin have done. Felten suggested spending at least a year so you can be productive. Felten also explained that for people starting out, there currently is no career path that leads from being a working technologist to senior advisor, and it would be good if that existed.

Dr. Felten left a lot of time for discussion.

Tony Dahbura (Johns Hopkins) observed an important paradox that appears in society, that the more information becomes accessible, the more uninformed people's behavior appears to be, perhaps because they are reluctant to say "I don't know." Felten said that experience has shown that having more information has made people better decision-makers on the whole, but wouldn't go as far as Dahbura had in saying it was actually harmful. People need to have skills to use that information. People also are attempting to confirm their beliefs, and it has likely always been that way. It's important to know how to turn information access into better decision-making.

Iulia Ion (Google) asked what people who don't have sabbaticals can do to get involved and share their own views. Felten suggested getting in touch with a policy-maker or the people on their staff and developing a contact point. The staff people who answer the phone or work in the office are there largely to work with constituents, and educating a staff member well might have a greater impact than talking directly to the decision-maker.

Greg Shannon (CMU) pointed out that some organizations have legislative affairs people. Then Greg asked what it was like to have someone listen to him. Felten had done a lot of due diligence before going into the job, and knew he had to work with the people in the FTC who knew how to work with Congress and other decision-makers. Felten said you have to socialize the idea you want to get across, and he worked early on to develop a rapport with FTC staff members. Government is designed to make things hard to do, the checks and balances put there to prevent abuse. It's closer to university politics than you might think, quipped Felten.

William McAlter (Johns Hopkins Applied Physics Lab) asked where Felten learned about working in government. Felten said he learned about this through his struggle with the DMCA and how it affected his research, and later through being an advisor on the Microsoft antitrust case. Felten said it is something you have to learn over time.

Joe Kiniry (Technical University of Denmark) said that, having worked in both Europe and America, he had discovered some differences. For example, in Denmark, there is not a single legislator educated in STEM. In America, that tastes different. Felten replied that having politicians trained in sciences is a good thing—for example, there's a New Jersey senator who was trained in physics and actually understands statistics. But in the House, that is actually quite a rare thing, and that becomes an issue. Part of that is the career gap. Another is the belief that knowledge of technology disqualifies you from participating in that policy discussion.

Bill Simpson thanked Felten for a great call to arms. He also suggested getting involved in campaigns, as he has done, by providing technical support. Simpson pointed out that of those people you are participating in campaigns with, about half of them will become staffers. Simpson said he has been doing this since the mid-'70s, and now visits people he knows in congressional offices when he visits Washington. Felten agreed that this is excellent advice, and went further by saying that campaigns have become much more analytical and data driven, so there is now a greater need for technical support, to apply your expertise to campaigns.

Chris Watjic (Georgetown) wondered how to help politicians identify quacks. Felten suggested helping people recognize what type of credentials represent expertise, such as being a long-time member of the IETF (like Bill Simpson), or being a program chair or program committee member. Unfortunately, sometimes credibility comes from a person who works for a company that has a stake in the outcome of a decision.

Michael Hicks (University of Maryland) asked whether there is a way that researchers could do their jobs better to help with the political process. Felten said that we currently focus on building knowledge brick-by- brick, but sometimes we need to choose our projects differently. Also, we need to examine how we decide to publicize our findings, which could be as simple as emailing a contact about your research.

There was much more discussion, as well as a lot more that Felten provided in his well-received and prepared talk. I suggest that you watch the video or listen to the audio on the USENIX Web site.

Network Security

Summarized by Gang Wang (gangw@cs.ucsb.edu)

Greystar: Fast and Accurate Detection of SMS Spam Numbers in Large Cellular Networks Using Gray Phone Space

Nan Jiang, University of Minnesota; Yu Jin and Ann Skudlark, AT&T Labs; Zhi-Li Zhang, University of Minnesota

Yu Jin talked about Greystar, their system for detecting SMS spam in cellular networks. The authors' key assumption is that spammers randomly select target phone numbers from a finite phone number space. So they will inevitably send messages to numbers that normal users typically would not reach: for example, those associated with laptop data cards or electricity meters. Yu called these numbers "grey" phone numbers.

Then Yu described their statistic model for SMS spammer detection based on grey phone numbers. To evaluate the system, they experimented with five-month SMS call records from AT&T. The experiments demonstrated that they could achieve high accuracy, and also detect spammers much faster than existing crowdsourced user reports. In particular, Yu mentioned that their system, once deployed, could reduce spam volume by 75% during peak hours.

Several people asked about the possibility of using other features to improve the system—for example, messages sent per day. Yu responded that these features were complementary, and some could easily raise false alarms. Another audience member asked what would happen if attackers didn't target randomly selected phone numbers but real, valid phone numbers collected via other methods, e.g., social engineering. Yu said that, based on their real-world data, 90% of the spammers fall into their assumption. Finally, there was a question about possible collaboration of different carriers to combat SMS spam together. Yu said collaborations would be definitely helpful, in their case, to accurately identify grey phone numbers and catch spammers. However, in practice, this type of collaboration was still very hard to achieve.

Practical Comprehensive Bounds on Surreptitious Communication over DNS

Vern Paxson, University of California, Berkeley, and International Computer Science Institute; Mihai Christodorescu, Qualcomm Research; Mobin Javed, University of California, Berkeley; Josyula Rao, Reiner Sailer, Douglas Lee Schales, and Marc Ph. Stoecklin, IBM Research; Kurt Thomas, University of California, Berkeley; Wietse Venema, IBM Research; Nicholas Weaver, International Computer Science Institute and University of California, San Diego

Wietse Venema presented their work on detecting stealth communication over DNS. Today, attackers can piggyback communication in DNS queries to secretly transmit information. Wietse presented a new measurement procedure that could bound the amount of information that a domain could receive through DNS queries. The key idea is to use lossless compression. Potentially, attackers may encode information in a DNS query name, query type, query timing, or a combination of them. The authors' procedure takes all potential information vectors and investigates the upper bound of information that can be encoded in a stream of DNS queries. Using this bound, they can narrow down surreptitious communications to a small set of DNS lookups. Also, the set should be small enough for manual assessment.

A practical challenge for this procedure is how to minimize the analysis burden in the face of tens of millions DNS lookups. In the talk, Wietse showed how they pare down the volume of DNS queries by eliminating obvious benign candidates. They evaluated this procedure with a real-world data set of 230 billion DNS lookups. Their procedure had no false positives and was able to detect 59 confirmed tunnels. Wietse also pointed out that they found that 4 KB/day was a reasonable threshold, which led to an acceptable assessment burden (one to two events per week) for enterprise sites to take in practice.

One audience member asked whether they could share the data set. Wietse said they were happy to share the code and results, but the data set was from IBM and could not be shared because of company policy. Another audience member asked whether this approach would still work if DNS queries were encrypted. Wietse's reply was positive. Someone asked how they determined the thresholds in the measurement procedure. Wietse said that the tradeoff was made based on their empirical analysis of real data: a smaller threshold (4 KB) for individual clients and a larger threshold (10 KB) for extremely aggregated logs.

Let Me Answer That for You: Exploiting Broadcast Information in Cellular Networks

Nico Golde, Kevin Redon, and Jean-Pierre Seifert, Technische University Berlin and Deutsche Telekom Innovation Laboratories

Kevin Redon presented a new attack in cellular networks. Focusing on GSM, he demonstrated how attackers could hijack a mobile terminated service (e.g., phone call) and perform a denial of service attack. This attack can occur because GSM initiates the paging procedure on a broadcast medium before setting up any authentication. So attackers who are also in this network can observe the paging requests of other phones (victims) and send a fake paging response on behalf of the victim. If the attacker responds faster than the victim, the GSM network will accept the fake response and ignore the victim's response. After these replies, GSM will set up service authentication (which will fail) and the victim's service will be dropped.

Kevin demonstrated the feasibility of this attack using freely modifiable software and hardware for GSM networks. It is worth noticing that other standards, such as UMTS or LTE, also have the same (vulnerable) paging procedure. At the end of the talk, Kevin showed a list of possible countermeasures, using A5/3 encryption to prevent hijacking, for example, or performing authentication before paging procedure, etc. Kevin said they notified the respective standards organizations about this problem but have had no immediate reaction from them so far.

Video about the attack can be found here: https://www.youtube. com/watch?v=oep3zpY6cvE, https://www.youtube.com/watch?v=4umb2P-93BQ.

One audience member asked which countermeasure is actually deployable in practice. Kevin said most countermeasures are about protocol modification, which requires efforts from standards organizations. At the very least, we could adopt the more secure A5/3 to mitigate the threat. A follow-up question was whether they tested any proposed countermeasures using their testbed. Kevin said they empirically tested a few, but not all of them. Another audience member asked whether the cellular tower could notice the presence of this attack based on the duplicated paging responses. Kevin said the cellular tower could detect that there were two phones sending responses, but could not tell which one was the legitimate one.

Potpourri

Summarized by Ziming Zhao (zzhao30@asu.edu)

Dowsing for Overflows: A Guided Fuzzer to Find Buffer Boundary Violations

Istvan Haller and Asia Slowinska, VU University Amsterdam; Matthias Neugschwandtner, Vienna University of Technology; Herbert Bos, VU University Amsterdam

Istvan started his presentation by explaining that buffer overflows are still among the top three threats after 40 years of research. He then provided context about state-of-the-art automated testing approaches by explaining static analysis and symbolic execution. Static analysis is difficult to make path-sensitive and inter-procedural, and it generates many false positive and negatives. Even though symbolic execution could achieve significant code coverage, the exponential number of possible paths means it is not practical in many cases.

By showing a piece of buggy code from the Nginx Web server, Istvan concluded that complete code coverage cannot even guarantee triggering a bug. To address these issues, Istvan and his co-authors tried to narrow down the scope of their research problem. Instead of pursuing complete coverage of paths, they focused on high-priority code fragments, especially the code that accesses an array in a loop.

They proposed to first identify and rank loops based on their bug probability, calculated from features such as whether the loop has a pointer dereference. Using taint tracking, they were then able to identify the variables that may influence potential buggy loops. Finally, they performed symbolic execution only on these identified variables, which reduces the test space tremendously.

To explain their symbolic execution approach, Istvan first laid out the basics of symbolic execution followed by some traditional search strategies, such as depth first search and code coverage. They proposed using a value coverage search strategy which showed incredible performance in terms of search time. In conclusion, Istvan showed that their implemented tool, Dowser, could detect bugs in less than a minute for some programs that previously had required over eight hours analysis.

Someone asked how their dynamic analysis was guaranteed to find the code that modifies pointers. Istvan answered that the learning process was important; more time spent on learning would increase the quality. Someone asked how the results of value coverage were searched without source code. Istvan replied that the only part of their analysis using source code was the static analysis to find loops. Someone asked which semantic engines was their work based on. Istvan replied they used some standard semantics engines which have been out for years.

MetaSymploit: Day-One Defense Against Script-Based Attacks with Security-Enhanced Symbolic Analysis

Ruowen Wang, Peng Ning, North Carolina State University; Tao Xie, University of Illinois at Urbana-Champagne; Quan Chen, North Carolina State University

Ruowen Wang started his presentation by introducing Metasploit, a Ruby-based penetration framework that contains more than 1000 attack scripts. The typical mechanism that a Metasploit script uses has four steps: it (1) probes a vulnerable target, (2) generates an attack payload dynamically based on the probe results, (3) sends that payload to the victim, and (4) triggers the vulnerability and compromises the target.

He then showed a number of Internet news articles about hackers using Metasploit to attack production systems; Metasploit as a powerful penetration tool has turned into a real-world weapon. Ruowen and his co-authors have proposed an effective technique to defend against attacks launched by Metasploit. He explained that their approach does not require a vulnerable applications and testing environment, but only uses security-enhanced symbolic analysis to generate IDS signatures.

Ruowen presented the architecture of their tool, MetaSymploit. MetaSymploit symbolically executes attack scripts collected from Metasploit and captures fine-grained attack behaviors and conditions. By using both symbolic values and concrete values in the generated payload from MetaSymploit, they were able to extract signature patterns for specific attack payloads.

To implement their idea, Ruowen presented their efforts to develop a symbolic execution engine for Ruby 1.9.3. They have integrated their work into Metasploit 4.4. Based on their evaluation, their tool could generate snort roles for 548 attack scripts. The performance required less than one minute for each script, impressive considering that symbolic execution was adopted.

Someone asked whether Ruowen had considered combining the generated rules. Ruowen replied that they are looking into some work on aggregating rules with regular expressions. Session chair David Wagner asked how hard it is for the bad guys to defend against this work. Ruowen said it is possible for bad guys to defend against their technique, but they face challenges. Shuo Chen (Microsoft Research) asked whether the input size of the symbolic execution introduced any performance issues. Ruowen replied that it was not an issue in their study.

Towards Automatic Software Lineage Inference

Jiyong Jang, Maverick Woo, and David Brumley, Carnegie Mellon University

Jiyong Jang explained the motivations for software lineage inference, which is to recover the lineage given a set of program binaries. Software lineage inference could provide information in many security scenarios, such as malware triage and software vulnerability tracking. Even though there are abundant analyses of software history and lineage, how to automatically infer software lineage from binaries is still an open question.

To address this problem, Jiyong presented a list of software features that could be utilized to infer a temporal ordering and evolutionary relationships among binaries. He also explained some features were chosen based on the common understanding that program size and complexity tend to increase rather than decrease as new revisions are released.

To measure the difference between the feature sets from binaries, Jiyong presented several techniques that include symmetric distance, dice coefficient distance, Jaccard distance, Jaccard containment distance, and weighted symmetric distance. Jiyong then showed that the lineage inference algorithm they proposed performed similarly regardless of the distance metrics, with Jaccard containment distance being the exception.

To evaluate their work, Jiyong presented some lineage examples from real-world binaries compared with ground truth generated from source code. Jiyong focused on one example where the automatically inferred lineage differed from the ground truth. Jiyong explained that a deeper manual analysis revealed that a version of the software was reverted to version 1 after several generations instead of evolving from the previous version. This was the root cause of the difference, and their automatically inferred results were accurate and able to identify this change.

Sumam Jana (UT Austin) asked whether their work is based on source code or binary. Jiyong replied their work only needs binaries. Someone asked about how they handled obfuscated malware. Jiyong replied they had considered a lot of metrics, including some dynamic features, and had combined them with other features to achieve better accuracy for malware. Someone from Maryland asked about the particular challenges involved in extracting lineage relationships from binaries since there is already work doing the same thing for source code. Jiyong said working on binaries required much more careful feature selection.

Mobile Security I

Summarized by John Scire (jscire@stevens.edu)

Securing Embedded User Interfaces: Android and Beyond

Franziska Roesner and Tadayoshi Kohno, University of Washington

Franziska described the motivation for their work on embedded user interfaces. Currently, Web browsers have a simplistic and mostly secure way of embedding third-party material into a Web site using iframes, which provide secure isolation between UI elements. However, Android does not have any way to do cross-application embedding securely. What currently exists in Android is the embedding of ads in an application, but this is done using third-party ad libraries. She gave a great example to demonstrate a type of attack that exists with these ad libraries on current stock Android, where an embedded ad could change all of the other child UI elements in an application. She went on to describe some of the previous work related to the embedded UI in Android, but these only involved approaches specifically tailored to these ad libraries. The approach that her team took was creating a modified version of Android that supports secure cross-application embedding, which they call LayerCake.

Franziska provided some background knowledge about how Android applications work so as to understand how their modification works. An Android application consists of one or more elements that are known as Activities and within each Activity there is a tree of UI elements known as Views. The modification itself, as described by Franziska, involves three components. The first is the separation of processes, which essentially works similarly to iframes. They created a new View called EmbeddedActivityView that will display the embedded content. This new addition allows the parent and child elements to be isolated from one another, while still having communication between them. The second component is to use separate windows for each of these Embedded Activity Views. This is because their first component, creating new Views, still allows for UI elements to grab data passing through the layout tree. The third component involves various other additions to handle other security concerns discussed in the paper.

The evaluation of LayerCake involved, in total, over 2500 changes that included fundamental changes to the ActivityManager and WindowManager. In the applications they tested, higher load times were required to load all of the embedded activities. The parent activity load times, on the other hand, were unaffected. Because each Activity is in its own window, the Android WindowManager has to be involved to switch focus based on user input. This additional indirection, however, had little impact on the application. For instructions on how to download and flash LayerCake onto an Android device, go to http://layercake.cs.washington.edu.

Will Enck (NC State) asked about how this modified Android would handle software dependencies with embedded UI elements in an application. Franziska replied that there was not one real answer, but she provided some approaches, including installing the dependencies at the Android store. Paul Pierce (UC Berkeley) asked about having any plans with Google to integrate this into stock Android. Franziska replied that she had not talked to Google about this yet, but would love to!

Automatic Mediation of Privacy-Sensitive Resource Access in Smartphone Applications

Benjamin Livshits and Jaeyeon Jung, Microsoft Research

Ben began his talk by providing an overview of permissions in mobile applications. Permissions mainly go under two categories: installation-time and runtime, these names describing the point at which they are shown and asked for. Installation-time permissions were not enough, however, because users simply click "Accept" and continue using the application without really knowing what they are consenting to. While this may have implications towards iOS and Android, the rest of the talk focused on location data permissions on the Windows Phone platform.

Ben explained an MS guideline document that has various criteria for properly obtaining a user's permission, which in the scenario of location data requires having some kind of prompt telling the user that an application wants to use her location. By looking at how various example applications implemented prompts, Ben and his team were able to come up with a static analysis approach using a Control Flow Graph to locate missing prompts for resources and put them in when they were actually missing. They developed two different methods to do this. The first was the Dominator-Based method, where the prompt would be placed at the dominating node for a particular access request node. Ben said that this method, although extremely fast, prompts the user long before the actual request, which was something that they wanted to avoid. The other method was Backward Placement, which works backwards through the graph, starting at the resource access and putting the prompt at nodes prior to these accesses. The problem with this approach is that you could have multiple placements of prompts for the same access.

The authors evaluated 100 applications with an average size of 7.3 MB and an average of two location accesses per application. The Dominator method was faster than Backward Placement and was also much more successful in terms of properly inserting missing prompts in applications. Taken together, these approaches were 91% successful and, for unique resource accesses, 95% successful in correctly placing missing prompts.

Rik Farrow asked why this approach wasn't just put into the OS itself. Ben said that this not only required a lot of "soul searching," but also a bit more than just simply placing it into the OS. He added that they also want to allow the developer to have some control as well. Someone asked about checking what the actual prompt says if it does exist within the application. Ben replied that they do not have any further analysis on the actual prompt text. The questioner said that you could build this into the OS by having mandatory text and then optional text with a particular prompt. Ben said that this was not impossible to do.

Flexible and Fine-Grained Mandatory Access Control on Android for Diverse Security and Privacy Policies

Sven Bugiel, Saarland University; Stephan Heuser, Fraunhofer SIT; Ahmad-Reza Sadeghi, Technische Universität Darmstadt and Center for Advanced Security Research Darmstadt

Sven started by briefly describing the current state of Android security, which has proven to be insufficient several times over using various attack vectors. Thus, better security mechanisms need to be in place. He introduced previous academic security extensions that have been developed, such as Saint, XManDroid, and SEAndroid. From these, Sven and his team made two key observations: (1) most of these extensions involved a form of mandatory access control that was modified to fit a specific problem and not a general fitting, and (2) access control on Android needs to be both on a user-space level and a kernel-space level. Sven mentioned a particular example of a rootkit bypassing a middleware enforcement mechanism altogether to access a particular service within Android. Using these two observations, Sven and his team came up with a general system-wide mandatory access control solution for Android called FlaskDroid.

FlaskDroid employs a policy language, SELinux to be specific, in order to perform the MAC enforcement policies. Along with this, it uses an object manager that allows processes or applications to be aware of the exact kind of data they are handling that includes attributes such as a particular security type for that object. Examples of the language were provided as further explanation, but there are a multitude of them in the paper. In terms of the system itself, FlaskDroid uses SEAndroid at the kernel layer of Android for low-level MAC and a middleware module at the user-layer. Both of these components sit behind the API for services on Android to control enforcement and are connected to the security servers for policy queries. Sven added that the user and application developer can add policy rules specific to the settings they want that will get updated on these servers. Then, to hook the two components together, they use a Boolean mechanism whereby both the user-layer MAC and kernel-layer MAC communicate.

Because this employs the SELinux policy language, one could argue that this might weigh down FlaskDroid with an overwhelming number of rules. As it turns out, Sven and his team produced vastly fewer rules than SELinux in FlaskDroid's current setup. He also showed some use-cases pertaining to how a sample application may utilize this new MAC mechanism. One example involved a phone dialing application where the user is presented with a dial pad. The user can then turn on a phone booth context, which is a sort of mode in SELinux, that will disable the ability to leave the dial pad screen entirely. This way a person using your phone to try to dial a phone number cannot use the phone to do anything else. The paper itself has many more use cases and the source code for FlaskDroid can be viewed at www.flaskdroid.org.

Rik Farrow asked about the ability of malicious applications to loosen the "everything denied by default" approach of SELinux. Sven replied that the policy set by an application is only for the application and cannot interfere with access to another application. Will Enck (NC State) asked about the choice of using SELinux in the implementation due to its unmanageability. Sven responded that the choice was primarily due to wanting to merge their implementation with SELinux. Sven stated that SELinux becomes unmanageable only because of the sheer number of rules, but for smartphones it was not nearly as bad. However, Sven said they could improve this if they did in fact choose a different language.

Invited Talk

Summarized by Rahul Pandita (rpandit@ncsu.edu)

Windows 8.1 Supporting User Confidence

Crispin Cowan, Senior Program Manager, Windows Core Security, Microsoft, Inc.

Cripin started the presentation by sharing with the audience his experience of a 2010 talk where he compared Windows security with UNIX security, and humorously admitted that he was a UNIX fan prior to working at Microsoft. In retrospect, he added, Windows security was just fine even then, but he pointed out that not only have attackers gotten better, but end users have become more demanding. These two factors have significantly increased the need for security in the operating system environment.

Crispin then dived deep into the features of Windows 8 directed towards boosting end-user confidence in the security of the operating system. He touched on a wide range of features, starting with hardware-based security, where he introduced Unified Extensible Firmware Interface (UEFI). UEFI is an improvement over the existing Basic Input/Output System (BIOS) to ensure that only a verified OS loader is used during boot time. This effectively addressed issues with malware that targets OS loaders. He then went into details about other security measures to ensure a safe boot of the OS in Windows 8.

Moving forward, he introduced the security features of interacting with the Windows App Store. He presented the feature called app container. An app container allows the OS to contain the effects of a rogue app installed by a user. App container also facilitates the seamless transfer of data with the OS (like opening a file) and the app by use of a mechanism Microsoft terms an authentic user gesture (AUG). The security principle behind the functioning of AUG is that the AUG can only be initiated by a user and not by an App. This was followed by a series of demos of AUG, mostly involving opening and storing a file within an app.

Crispin also presented the concept of a kill bit (reminds me of a kill switch) in apps. Having the kill bit in place allows Microsoft to remove a rogue app from all the devices remotely. He assured us that such a capability is used sparingly and after careful evaluation of the app that needs to be removed. He also explained that every app that is installed on Windows 8 has to be digitally signed by the developer and has to be installed only through the Windows App Store.

Among other features, he talked about modernized access control. In particular, he presented new sign-in options in Windows 8, including pin, passwords, picture passwords, access cards, and even biometric verification support. He proceeded to show a demo of the picture passwords but could not show it in action due to screen resolution issues of his Windows 8 device when connected to the projector for the talk. He concluded his talk by reiterating some of the core security features of Windows 8.

Felix "FX" Lindner (Recurity Labs) asked why Microsoft delegated the task of issuing and managing certificates for OS Loader in UEFI to a third party. Crispin responded that certificate authorities (CA) were a well established business and outside the scope of Microsoft's business interests. Furthermore, he said that existing certificate authorities were doing a great job, and thus Microsoft did not feel the need to manage certificates on their own.

Someone followed up by asking, what if the CA itself was compromised? Crispin said that there was a kill-bit built right into the UEFI module to remotely disable it.

Two people asked about the kill-bit and expressed their concerns about abusing them. Crispin addressed their concerns by assuring them that Microsoft carefully weighs its options before using the kill-bit and that extra carefulness is required because abusing kill-bits also has legal implications.

Someone followed up by asking, what if a security researcher wanted to keep a malicious app for experimenting on it? Crispin clarified that the kill-bit was mandatory and not optional and so, if exercised by Microsoft, the malicious application had to go. He hinted, however, that there were some indirect workarounds if someone wanted to keep a malicious app.

Session chair Wenke Lee(Georgia Tech) asked whether the Surface RT—the first device that ships with Windows 8—is locked into the Windows App Store. Crispin affirmed this. Lee further inquired how difficult it is, given the safety features of the Windows 8, for students to write and install their own Apps. Crispin humorously responded that "students might have to jump some hoops to do that."

Applied Crypto I

Summarized by Bhushan Jain (bpjain@cs.stonybrook.edu)

Proactively Accountable Anonymous Messaging in Verdict

Henry Corrigan-Gibbs, David Isaac Wolinsky, and Bryan Ford, Yale University

Henry Corrigan-Gibbs outlined the motivation for their work on an anonymity system called Verdict by presenting a scenario: an activist in a country X learns about a prime minister's stash of stolen money and wants to anonymously publish this information before the elections next day. Henry then took us through the options available to the activist based on existing systems and pointed out possible attacks to expose the activist or delay message posting. An onion routing solution can be broken by a state-owned ISP using a first-last correlation attack. Henry then introduced dining cryptographers networks (DC-nets), anonymous communication networks resistant to traffic analysis in which a group of people contribute an equal length message to derive one single anonymous message at the end of the protocol. Dissent is a practical implementation of DC-nets.; however, the prime minister's supporters can infiltrate the group and cause denial of service attacks on the Dissent system. The shuffle protocol used by Dissent to assign blame to the disruptor takes time and so the PM's supporters can postpone the posting of an anonymous message until after the elections are over. If the disruptors can control 10% of the nodes involved in the protocol, they can block communication for a day or more.

Verdict is a system derived from Dissent to leverage the traffic analysis resistance and scalability of Dissent but with lower blame cost. The main idea is that the group members have to prove that the message they are sending is correctly formed. Thus, Verdict identifies the disruptors before they launch the denial of service attack.

Henry then took us through design challenges and optimizations to make the system fast. Verdict resists traffic analysis attacks by having each client transmit an equal length cryptographically indistinguishable message per round. In order to make the sender's transmission indistinguishable, every other client sends a dummy message encrypted using an ElGamal-like scheme while the sender sends the original message encrypted in the same format. In order to prove that their transmissions are well formed, the clients attach non-interactive zero-knowledge (NIZK) proofs of knowledge to their ciphertexts. He explained optimizations to improve performance in case of long messages, lazy proof verification, and a hybrid Dissent+Verdict DC-net.

The fastest implementation of Verdict provides a 5.6x speedup over existing systems and the hybrid Dissent+Verdict implementation gives 138x speedup. In a 1024-node cluster, the lazy Verdict optimization reduces messaging latency by 2.3x over pure Verdict, and the hybrid version reduces latency by 27x. The pure Verdict version can reduce the cost of finding disruptors from Dissent by about 200x.

When asked why not use the provable shuffle anonymity system instead of Dissent to relay messages, Henry said that Verdict achieves better efficiency for messages of varying length or multiple rounds over using provable shuffle. Someone asked whether the provable shuffle to assign slots can be replaced by a rotation. Henry said that it would work but may not be any faster. When asked if the hybrid version may take more time due to disruptions, Henry agreed that it takes time to switch to Verdict from Dissent in case of disruptions and that the hybrid version trades off the performance in the general case with the performance during disruptions.

ZQL: A Compiler for Privacy-Preserving Data Processing

Cédric Fournet, Markulf Kohlweiss, and George Danezis, Microsoft Research; Zhengqin Luo, MSR-INRIA Joint Centre

Cédric Fournet presented their work on a compiler for data processing with strong privacy guarantees. He started by explaining the need for privacy preserving data processing using examples of smart meters and pay-how-you-drive insurance. The main argument is that the service provider doesn't need to know all the details of usage as long as the provider is getting paid the correct amount. The existing cryptographic solutions need intervention from security experts every time the policy or query is changed. To solve this problem, Cédric introduced ZQL, a high-level language for querying data together with its query compiler that synthesizes cryptographic protocols from a source definition to generate code that can run on various platforms.

ZQL supports a subset of F# language and iterators on data tables. ZQL can compute math functions, exponentiation, and table lookup operations while operating on secrets. ZQL uses a combination of Pedersen commitments, NZIK arguments, and CL-signatures for cryptographic implementation. One limitation of ZQL is that the intermediate result structure has to be public even though contents in that structure are private. The ZQL compiler takes the data specification and query as input and generates queries for the parties involved to be used in a cryptographic protocol. A F# or C generator then consumes these queries and outputs reference implementation in F# or C.

They extended ZQL to support cryptographic primitives like long integer, exponents, hashes, signatures, and commitments. Now, ZQL generates an extended query from source query using a compositional shared translation by inserting commitments, openings, and proof assertions. The extended query is subject to code specialization to generate a NIZK proof of knowledge. They also use the extended query to generate a simulator to reason about privacy and an extractor to reason about soundness.

Cédric then demonstrated the system for two sample computations and their verification. He showed how a verifier can verify that the value x+y is computed correctly. He also showed how the protocol works in the case of a pay-how-you-drive query. The cryptographic evidence is linear in size as compared to the computation. The verification proof does not contain any information about the input but provides computational integrity. The system was evaluated using RSA 1024, RSA 2048, and pairing-based crypto. The proof size is a few KB for the test cases.

When asked about the different tradeoffs for one of the related works, Pinocchio, Cédric mentioned that Pinocchio proofs are constant size and the verifier computations are small but the prover has to do more work. Pinocchio may be preferred for computation-intensive processing for a limited amount of data while ZQL will do better for large amounts of data processing. Table lookups that are very important for ZQL cannot be done using Pinocchio. However, Cédric et al. are looking at how to combine the two. The code will be available very soon.

DupLESS: Server-Aided Encryption for Deduplicated Storage

Mihir Bellare and Sriram Keelveedhi, University of California, San Diego; Thomas Ristenpart, University of Wisconsin-Madison

Sriram Keelveedhi presented a system called DupLESS, a server-aided encryption system for deduplicated storage. Deduplication saves storage resources by avoiding storing duplicate copies of the same file. Their goal is to securely deduplicate in the presence of an untrusted storage service and to provide client compromise resilience. DupLESS trades off the storage savings and performance efficiency of plaintext dedup with increased security and compromise resilience for client files. He explained how existing solutions either do not allow deduplication or are not resilient to client compromise. Even the convergent encryption solution that achieves deduplication and compromise resilience is vulnerable to brute-force attack by the storage provider to recover the original file.

DupLESS uses server-aided encryption by leveraging a key server that helps clients encrypt their files. Every client sends the key server a hash of the file, and the key server computes the key to be used to encrypt the file using a PRF on this hash value. All clients encrypt their files with the same key K and send this encrypted file along with the encryption of the key K under their own key. The first file is deduplicated as it is the same for all clients. The second file is small enough that even though it is not deduplicated, the overhead isn't much. This scheme falls short when a strong adversary can compromise the key server and leak the key K used to encrypt all copies of that file. They implemented an oblivious PRF protocol between the key server and the client to defend against this attacker. This protocol is optimized to use sessions between the client and the key server, and the actual OPRF query takes a single round as authentication is done only during session establishment. On evaluating this protocol on EC2, they observed that the protocol performance was close to round trip time for the optimized version.

Sriram then explained the details of the DupLESS system design, which uses a storage service that provides a set of APIs to manipulate files. He then took us through the translation of a storage put query to steps for DupLESS. Put and get were the most expensive operations for DupLESS. A put operation takes 16% extra time to upload a file and increases the size by about 10%. DupLESS costs 4.4% extra space as compared to plaintext deduplication. In future, DupLESS may support keyword search, complex file systems, and heuristics on which files to select for deduplication.

Indranil Banerjee (Qualcomm) asked what secure means in the context of deduplication. Sriram replied that security implies semantic security and no information leakage about the data. A follow up question was how does deduplication increase the risk of compromising confidentiality. It is difficult to combine encryption and deduplication as seen in existing solutions, and DupLESS provides a solution to mitigate risks of attacks against these solutions. Someone asked if a key server can do brute-force attacks on the file if the key server is compromised. Sriram replied that this attack is possible only if the key server can monitor the network traffic to get the ciphertext. Does the implementation have to take into account the backend storage provider? As long as the storage provider exposes APIs as discussed, DupLESS is seamless to the implementation behind the scenes. Zack Peterson asked why couldn't an encrypting proxy perform all the computations instead of the client. The encryption proxy becomes the natural target for the attacker, answered Sriram. With Dupless, even if the keyserver is compromised, they at least have guarantees of a convergent encryption. David Jacobson (Qualcomm) asked if a side channel could leak information that the file already existed on the storage server based on the time required to store the file. Sriram said that the information that is leaked is based on the location of deduplication. Deduplication on the storage provider side will force transmission of the whole file irrespective of whether the file already existed on the storage server. Someone asked why not use DTLS instead of the OPRF protocol. Sriram replied that the OPRF protocol can be derived by tweaking the DTLS protocol.

Large-Scale Systems Security I

Summarized by Gang Wang (gangw@cs.ucsb.edu)

Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse

Kurt Thomas, University of California, Berkeley, and Twitter; Damon McCoy, George Mason University; Chris Grier, University of California, Berkeley, and International Computer Science Institute; Alek Kolcz, Twitter; Vern Paxson, University of California, Berkeley, and International Computer Science Institute

Kurt Thomas presented their study on underground markets that trade fake Twitter accounts. To understand this problem, they monitored 27 account merchants over 10 months and purchased 100k fake Twitter accounts from them. Kurt said they found these merchants were using many sophisticated methods to circumvent automated account creation barriers. For example, account merchants used crowdsourcing services to solve CAPTCHAs, collected fraudulent email credentials from Hotmail and Yahoo, and also used tens of thousands of IPs (proxies, VPNs) all over the world to evade IP blacklisting.

To detect these auto-generated accounts, they developed a classifier, which looked at patterns in naming conventions and features that indicate automated accounts registration (e.g., events sequence triggered during signup and timing). With the help of Twitter, they scanned all Twitter accounts registered last year and found several million fake accounts. According to Kurt, the revenue of these account merchants are about $100,000 to $400,000. After Twitter adopted this technique, many account merchants started to go out of business. During the talk, Kurt even showed screenshots of some merchants' announcements, saying that they could no longer provide the service due to unknown changes in Twitter, which is impressive.

Someone asked about possible evasions of the proposed classifier. Kurt commented that account merchants may get around the naming features, but it was still hard for them to deal with features indicating automated account registration. Another audience member asked about the acceptable false positive rate for Twitter. Kurt said he did not know because this was confidential for Twitter. Someone asked whether these fake accounts include compromised accounts. Kurt said the markets they focused on were mainly selling automatically registered accounts, but there are merchants who sell compromised accounts. Another person asked whether they monitored the price change over time. Kurt said that the prices in the markets they monitored were relatively flat. Someone asked whether these merchants would resell those accounts. Kurt confirmed that certain merchants did scam their customers: after selling the accounts, the merchants would try to secretly retrieve the accounts back and resell them to other customers.

Impression Fraud in Online Advertising via Pay-Per-View Networks

Kevin Springborn, Broadcast Interactive Media; Paul Barford, Broadcast Interactive Media and University of Wisconsin—Madison

Kevin Springborn talked about their measurement study on impression fraud in online advertising. In regular online advertising, advertisers place advertisements on publishers' Web sites and pay publishers based on how many users have viewed the ads (impressions). In the talk, Kevin described pay-per-view (PPV) networks that help dishonest publishers to drive traffics to their Web sites. PPV networks usually consist of compromised Web sites that render publishers' pages hidden in requested pages to users' browsers. In this way, they can generate additional, fraudulent impressions on publishers' pages. The true victims are advertisers who have to pay dishonest publishers for those impressions.

Kevin described their measurement approach. Basically, they set up three Web sites as honeypots, and then purchased traffic (addition impressions) from 34 traffic generation services who owned PPV networks. Surprisingly, they found those pay-per-view networks were rarely blocked by public blacklists and only had modest IP reuse. In addition, there was zero user-interaction from the purchased traffic. According to Kevin, the estimated fraudulent impressions delivered by PPV networks can reach as much as 500M per day, making this a multi-hundred-million dollar business. Kevin also pointed out some possible countermeasures, such as detecting zero-sized frames and blocking known PPV hosts.

One audience member asked whether all sites in pay-per-view networks were high-quality sites. Kevin answered that the quality level may vary from service to service. Someone asked about the click-through rate of these ads. Kevin said the ads were not actually "displayed." They were usually hidden in a zero-sized frame that users cannot see. So there were no user clicks generated. Finally, someone asked about the effectiveness of the countermeasures. Kevin said the countermeasure was easy to deploy and should be effective. But many current sites did not bother to do that, because they didn't have the incentive (they were not the victims).

The Velocity of Censorship: High-Fidelity Detection of Microblog Post Deletions

Tao Zhu, Independent Researcher; David Phipps, Bowdoin College; Adam Pridgen, Rice University; Jedidiah R. Crandall, University of New Mexico; Dan S. Wallach, Rice University

Tao Zhu talked about their measurement efforts to understand censorship in Chinese microblogging sites. Their focus was Weibo, the largest microblogging site in China. Because of censorship, people's posts (i.e., tweets) on Weibo would be deleted if the content were considered to be politically sensitive. The key question Tao wanted to explore was how fast the content deletion happened and possible mechanisms Weibo used to carry out censorship.

To collect the deleted (censored) Weibo posts, Tao focused on a set of sensitive users (several thousands) and crawled their timeline every minute over a two-month period in 2012. Tao found that Weibo was surprisingly fast in identifying and deleting sensitive posts. Most deletion happened within the first hour after the content was posted on Weibo. Tao said they tried to reverse-engineer the possible mechanisms Weibo used to achieve fast censorship. According to Tao, Weibo seemed to be using a keyword-based filter, combined with dedicated human censors. Also Weibo paid closer attention to users who frequently posted sensitive content.

One audience member pointed out that Weibo could potentially pollute Tao's data by intentionally returning incorrect timeline data. Tao said at the time of their study, they ran some validation tests by comparing the content returned from the API and the Web site, and did not find any inconsistencies. Another person asked how they knew the deleted posts were caused by censorship, not other reasons like spam or even self-deletion. Tao said the error message for self-deleted posts and Weibo-deleted posts were different. Also those users they monitored were carefully selected to make sure they were involved in censored discussion before. Thus their content was unlikely to be spam.

Another questioner asked what people would do after they got censored. Tao said he saw people started to perform some obfuscation on their posts, changing the form of keywords, for example, or using keyword substitutions. Someone asked how Weibo censors knew what topics to censor. Tao said there were multiple possible channels: they might get orders from the government to censor certain topics, or based on those sensitive users' recent posts or external resources like oversea news.

Thursday Keynote Address: The White House's Priorities for Cybersecurity

Andy Ozment, Senior Director for Cybersecurity, White House
Summarized by Theodore Book (theodorebook@gmail.com)

Andy Ozment spoke about the Obama administration's priorities for cybersecurity. He identified five basic priorities: protecting critical infrastructure (see Executive Order on Cybersecurity, below), securing the government, engaging internationally, improving incident response, and shaping the future. He emphasized the recent executive order on cybersecurity and summarized its main points.

Securing the Government: The federal government is a large institution with an unknown number of machines, people, and agencies. Work to secure it is being conducted by establishing standards and holding people accountable. There are a series of cross-agency priority goals: First, implementing trusted Internet connections. Currently, they don't know where they are connected to the Internet. They have found tens of thousands of connections, and are finding more all the time, but want to move to around 50. Secondly, they want to implement two-factor authentication, through the use of a smart card that provides both physical and electronic access. The third goal is continuous monitoring. Here, they seek to measure how secure they are—knowing vulnerabilities, and incentivizing higher security.

Engage Internationally: There need to be consequences for those who are trying to intrude, otherwise they will eventually get in. By using the word "intrude" rather than "attack," they are consciously using the language of espionage and not war. This process is really, really slow. They are engaging with the Chinese government, by raising this issue through diplomatic channels and a working group. They are trying to convey that there is a norm of behavior for espionage that distinguishes economic from government espionage. They want to discourage economic espionage by state actors. They are also working with the Russians in a long series of negotiations that have led to a red phone for cybersecurity incidents.

Improve Incident Response: A year ago, they held a national-level exercise on cybersecurity (these have traditionally focused on physical events like earthquakes and hurricanes). They have also been facing a steady year of DoS attacks against the financial services sector. They collected a list of attacked IPs and passed them to ISPs. Originally the process would take two weeks. They can now do it in minutes or hours.

Shape the Future: Attackers have the edge—they can keep trying until they succeed. They want to make things better by focusing on DNSSEC, routing security, building a cybersecurity workforce, and R&D into less vulnerable systems.

Executive Order On Cybersecurity
The recent executive order on cybersecurity has four goals: information sharing, privacy and civil liberties, standards, and the identification of critical infrastructure.

Information Sharing: The administration wants to have parties share information on attacks, so that it becomes possible to understand the scale of a single intruder's activity. They also want to share indications of intrusions, so that if an intruder is caught in one place, he will be caught everywhere. The current goal is for the government to share information with the private sector, not because the government necessarily knows more than the private sector, but because it is easier within current laws. They also want to change government culture and classify less data. The problem is that sharing information can cause that information to lose its value. Even limited releases of information are quickly picked up by adversaries. Still, they are going to share more.

They want to offer an intrusion detection system called Enhanced Cybersecurity Services that uses classified signatures. These signatures are given to private sector enterprises who are certified to store it and who have personnel with security clearances to handle it. A generic infrastructure firm can then run traffic through this black box to block malicious traffic. This is useful for small firms that don't have the in-house capability to analyze malicious traffic.

Privacy and Civil Liberties: Sharing government information with the private sector includes some privacy risks. Recent documents reference the Fair Information Practice Principles, which represent the accepted best practices for these questions.

Standards: Many companies have very poor cybersecurity standards. To improve this, the government is asking companies to share lessons learned from NIST. The goal is to build a framework (not a new set of standards) that collects standards together to provide a comprehensive guide for information security. These standards can become a basis for regulation. For example, regulators of existing industries (such as utilities) will be encouraged to create new regulations to force people to do what the government wants based on these standards.

Identification of Critical Infrastructure: There have already been many efforts (post Sept. 11) to identify critical infrastructure; however, they had more emphasis on physical threats. The goal of the current survey is to identify infrastructure, vulnerable to cyberattacks, whose loss would cause a catastrophic impact. This produces a shorter list that is easier to manage. It also allows government to prioritize companies for regulation and support. They are currently informing companies who made the list.

Legislative Priorities

The Obama administration is looking for more power from Congress to impose its views on cybersecurity. They dislike the idea of allowing states to make their own regulations, and prefer to concentrate power in the executive branch of the federal government so that we won't have 50 different sets of regulations. They want to collect more information from companies, but they also want to ban companies from providing information with personally identifying information, and to restrict the use of information collected by the government in that way.

A number of individuals asked questions after the presentation. Several individuals asked questions relating to the relative value of voluntary or mandatory standards. In reply, Dr. Ozment stated that there is no appetite for mandatory standards. People feel that a top-down approach would be harmful rather than helpful. They are then asking regulators to look at the voluntary standards, which they may then impose through regulation. A past attempt to get Congress to pass a law enabling the administration to impose standards failed. He also indicated that there are some problems with FISMA (the existing standards, which the questioner had criticized). Some organizations do a great job within this framework. Others have devolved into a regulation-compliance approach. Some problems are with the law, other with procedures required by the executive branch. They are trying to update it, and update their internal procedures.

In May 2011, they did not propose specific regulations, but the authority to regulate. The executive order is a good alternative solution. It allows for cooperative development of the framework which can then be imposed on many companies through existing regulators, although there will be some holes. Regarding other companies, three agencies were tasked to produce reports suggesting how they might be forced to act according to the administration's desires. They looked at various incentives and came up with nine. Some were: using the insurance industry by making the standards a possible factor in setting rates; using the rate recovery mechanism—regulated utilities could charge more to cover security expenses; prioritizing government grants and assistance; etc. Some of these will have to wait until the framework is done.

When asked if the proposal to provide classified signatures to certain providers was potentially anticompetitive, Dr. Ozment replied that this program has been piloted with the defense industrial base. There is no limitation on service providers. Anyone who is willing to meet the required standards and provide staff that can clear the background checks can take part.

On the question of whether incident reports should be publicly available, he indicated that we are in a difficult spot. Most companies do not report intrusions. We have to incentivize reporting. This means that a company should see a positive outcome as a result of reporting (e.g., intruder caught). Also, there should not be a significant downside, and for most companies, releasing the reports would be a downside.

In response to a wide-ranging question, Dr. Ozment stated the following: regarding [FDA] regulations prohibiting updates, some areas have a strong culture of safety and reliability that clashes with the culture of security—they prefer not to update. Regarding funding, no budget has been passed since Obama took office. Nonetheless, he believes that cybersecurity funding has increased—he will check on that. The government doesn't have a good way of tracking what it is spending money on, so there is no way for the administration to know what it is spending on cybersecurity. They are trying to figure that out by putting more regulations on government departments and requiring them to report more information. Regarding the recent unauthorized disclosures and what people are reading in the newspapers, he doesn't know what is going on, and could only read talking points in any case. He does want people to be able to trust the government and share their information with the government.

On the question of metrics, he observed that good metrics in cybersecurity are hard to come by. Right now, there is an obvious problem even without metrics. They will deal with metrics when the big things are tackled. On the question of privacy regarding biometric data, he indicated that society needs to define these issues, not just the government. Government can record that consensus. The commerce department released a "green paper" on privacy, which might be worth looking at. When asked about international engagement with allied and neutral countries, Dr. Ozment replied that the administration is helping other countries to develop norms of behavior as to what is acceptable in cyberspace.

Finally, on the question of education, he stated that they have national cybersecurity information month. Most is focused on universities, some on broad national awareness. They can generally raise awareness—it is more tricky to offer useful advice to individuals.

Large-Scale Systems Security II

Summarized by Frank Imeson (fcimeson@gmail.com)

You Are How You Click: Clickstream Analysis for Sybil Detection

Gang Wang and Tristan Konolige, University of California, Santa Barbara; Christo Wilson, Northeastern University; Xiao Wang, Renren Inc.; Haitao Zheng and Ben Y. Zhao, University of California, Santa Barbara

Gang Wang explained that a Sybil is a fake identity owned by an adversary and is maliciously controlled. Sybils have infiltrated social networks in a big way with 14.3 million on Facebook and 20 million on Twitter. The types of attacks Sybils can execute range from spamming unwanted advertisements, malware, phishing, stealing user information, and even political lobbying efforts have been made to try to release fake headlines about

Obama. One might assume that a Sybil's friends list would consist of mostly other Sybils, but this is not the case and in fact it is often the case that by the time a Sybil requests you as a friend they already have 20 or more of your friends in common with you, which on the surface makes them seem more legit to you and even to a static analysis of the graph.

Wang et al. proposed an alternative to static graph analysis which is to monitor the time and click events to distinguish between normal and Sybil users. This is motivated by the intuition that a Sybil is goal oriented and time limited so one might expect a pattern and efficiency to a Sybil's clicks. They investigate this approach by building a classifier that takes the clickstream (click time and events) as input and is trained offline with ground truth or trained online with input from a set of trusted users. Results of the classifier trained with ground truth only had 3% false negatives and 1% false positives. They shipped their software to Linkedin and Renren, where Linkedin trained the classifier with a ground-truth set of 40k users' clickstreams and was able to flag 200 new Sybils. Renren used the classifier on 1M users, flagged 22k suspicious users, and identified a new attack (embedded URLs in images). Wang concluded by stating that good Sybil detectors force the Sybils to slow down their click speed, mimic normal users and thus turn a beast into a puppy.

Siddharth Garg, University of Waterloo, asked how they cluster the graph? Wang answered that they use automatic clustering and just need to choose the resolution—too small and there's a loss of generality, too large and they lose accuracy. Garg asked how this software is effective over different data sets. Wouldn't it have to be unsupervised since there is no ground truth? Wang said that for this case we would need to generate a small data set to use for ground truth. Simon Chung (Georgia Tech) said that if the Sybil must limit its click speed, can it achieve the same throughput with many parallel Sybils? Wang answered that this is possible and is also why they do not simply classify Sybils by the time intervals between clicks, but also look at event transitions.

Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness

Devdatta Akhawe, University of California, Berkeley; Adrienne Porter Felt, Google, Inc.

Devdatta Akhawe began by explaining that this study was conducted on data collected from Google's Chrome and Mozilla's Firefox from users who have opted in to sharing "Telemetry" data. The information about how the user responds to the warning is recorded in the browser and shared with Google or Mozilla. The study "Bridging the Gap in Computer Security Warnings," Bravo-Lillo 2011, states that "most people don't read computer warnings, don't understand them, or simply don't heed them." However, since this was contradictory to Akhawe et al. findings, they conjectured that the original studies got these results because they were conducted in a lab environment, used trusted computers, presented the user with text-only warnings, and only required one click confirmation. Today's warnings are more engaging, including pictures, offering lay content with a link to read more details, and often requiring a multi-step override such as asking: are you really really sure?

Click-through rate is the ratio of warnings ignored over warnings shown, and they claim that an ideal click-through rate is 0% (all warnings should be heeded). This ideal rate should motivate content providers to fix their Web content in the case of false positives and thus would also alleviate users of annoying false warnings. The results show an interesting difference between Firefox, Chrome, Windows, Mac OS and Linux users. For example, Firefox had a lower click-through rate on both phishing and malware. Differences like this could be due to the amount of effort (number of clicks) it takes the user to ignore the warning but in the case of Firefox it only takes one click to ignore compared to Chrome's two clicks to ignore. Linux users also show a much higher click-through rate than Windows or Mac OS users. Also users of the beta or dev releases of the browsers show higher click-through rates. Which begs the question: "Does a greater degree of technical skill correspond to reduced risk aversion?" Akhawe states that this data shows that users do actually heed warnings, but the design does impact the users' behavior.

Frank Imeson (University of Waterloo) asked if there are times when a warning should be ignored and, if so, wouldn't that make the ideal click through rate non zero? Akhawe said that if there are false positives then the browser should ignore them and/or the content provider should fix their content, but this is a very long argument to be discussed more offline. Someone else commented that improvements are the result of improved warnings and an increase in public education. Is there a way to tease out the effects of education from the results? Akhawe said he doesn't know how they could do that but it would be useful information. Someone else asked whether there was a way to assess false positive rates. Akhawe replied not at the moment.

An Empirical Study of Vulnerability Rewards Programs

Matthew Finifter, Devdatta Akhawe, and David Wagner, University of California, Berkeley

Akhawe stayed on stage to present his work on reward programs for finding bugs. Google and Mozilla both offer a reward-based program to users who sign up to find bugs for their browser software. This study analyzes the difference between the traditional approach of hiring an engineer to find bugs compared to outsourcing this task to willing and able end users. If the user is able to find a bug, he or she is rewarded. This reward may be proportional to the severity of the bug as with Google; sometimes Google also revisits the severity assessment of the bug and, if they think the bug was more important than they originally thought, retroactively award more money to that user.

The finding states that vulnerability reward programs (VRP) are cost effective: Google spends about $485 per day while Mozilla spends about $658 per day, which is comparable to an engineer's salary doing the same job. VRPs have found on average more bugs than internal engineers for both non-critical and critical bugs: Mozilla reports that its VRP found 148 bugs compared to the 48 bugs found by the internal engineers, while Google reports that more critical bugs were found by VRPs than by their internal engineers. Akhawe also showed that Chrome has a smaller proportion of bugs considered critical than Firefox, which he hypothesizes to be because of privilege separation in Chrome. Akhawe concluded that Chrome and its VRP is more popular than Firefox. Google's VRP finds more bugs, has a shorter time to patch than Mozilla's, and has shown good repeat participation by users.

Someone from the University of Maryland complimented Akhawe on his talk and asked how VRP compares with black market reward programs. Akhawe responded that although black market rewards are higher, the required commitment is greater since they are looking for a working exploit. He added that people are generally good and black markets do attract the majority of bug finders. Jason Jones (Airborne Networks) asked about the effect of having programs like ZDI buying up exploits for Chrome and Firefox with respect to this work. Akhawe said that he doesn't know enough about ZDI but it would be interesting to take a look at. Someone else asked whether he had any data on false positives and had the time wasted on these cases been factored into the cost-effectiveness of VRP. Akhawe replied that he didn't have any data on false positives but in his conversations with Mozilla and Google no one had ever mentioned false positives as an issue. Jerry Tyson (Facebook) asked how this could work for Web apps and what the differences would be. Akhawe said that he has thought about it, thinks there would be advantages and disadvantages, and would love to get his hands on data from Facebook. Tim Fraser (DARPA) said that assigning metrics for measuring security is hard, but would the amount of money spent on the black market for these bugs be a good metric? Akhawe replied that black market money might be indicative but that metrics are difficult; the lack of spending by a vendor on bugs, however, may indicate a lack of security.

Applied Crypto II

Summarized by John Scire (jscire@stevens.edu)

Secure Outsourced Garbled Circuit Evaluation for Mobile Devices

Henry Carter, Georgia Institute of Technology; Benjamin Mood, University of Oregon; Patrick Traynor, Georgia Institute of Technology; Kevin Butler, University of Oregon

Henry began his presentation by discussing the current abilities of smartphones to perform SMC, or secure-multiparty computation. SMC involves two or more parties trying to securely evaluate some function without revealing their inputs. Smartphones now are very limited in several aspects, one of which is computational power, which SMC heavily requires. This is mainly due to the large amount of computation and memory necessary for garbled circuits, which are circuits constructed to perform the evaluation of an SMC function and whose inputs at each gate is obfuscated in some way. To solve this problem, Henry and his team devised a protocol that would push most of this heavy computation to the cloud, specifically in the two-party scenario, in a way that also allows all parties to be assured of the correctness and validity of the output.

The protocol uses Kreuter et al.'s maliciously secure SMC technique along with consistency checks and an outsourced oblivious transfer mechanism. To further describe the protocol, Henry provided the following scenario (featuring Alice, a Web server, Bob, and the cloud): (1) the construction of circuits by Bob, (2) an outsourced oblivious transfer involving all three parties to generate key information as well as Alice generating her garbled input, (3) the generation of Bob's input, (4) the evaluation of circuits by the cloud, and finally (5) the delivery of output. Henry mentioned that these steps retain all of the security checks used in Kreuter et al.'s previous work, but the formal proofs of security for the whole protocol are in their technical report, which is cited in the paper.

To test this protocol, Henry and his team put Kreuter et al.'s work onto servers and had a Galaxy Nexus phone connected to these servers. They then created a bunch of test mobile applications that use classic SMC functions, such as the Millionaires' Problem and edit distance, and ran these applications with and without the help of the servers. As a result, they saw that smaller inputs actually ran better on the device by itself, but of course larger inputs were dramatically slower on just the mobile device. The addition of the cloud performing the computation introduced a 98.9% speedup in terms of total execution time over just using the mobile device in the edit distance application with an input size of 128.

Someone asked whether anything would actually be problematic with Alice colluding with the cloud. Henry responded that allowing Alice and the cloud to collude could break some of the consistency checks that are in the protocol, which would cause Bob to lose assurance of the protocol. He also said that this is something that they could work on to improve.

On the Security of RC4 in TLS

Nadhem AlFardan, Royal Holloway, University of London; Daniel J. Bernstein, University of Illinois at Chicago and Technische Universiteit Eindhoven; Kenneth G. Paterson, Bertram Poettering, and Jacob C.N. Schuldt, Royal Holloway, University of London

Jacob first presented a brief introduction to TLS, which is used widely today for secure HTTP connections, and the RC4 stream cipher. Transport Layer Security, or TLS, consists of two protocols: the Handshake protocol and the Record protocol. The

Handshake protocol is used to establish the connection, whereas the Record protocol deals with the encryption of the payload of the packet. This research, however, only deals with the Record protocol because this is where RC4 is used. RC4 involves two algorithms: key scheduling and key generation. Key scheduling initializes a byte permutation using a key. Key generation then further permutes this byte permutation to create the keystream used for encryption. As Jacob mentioned, RC4 is used in more than 50% of all HTTPS connections, despite known statistical weaknesses. Using these known weaknesses, Jacob and his team created two plaintext-recovery attacks against RC4.

The first attack Jacob and his team made uses single-byte biases that exist in the first 256 bytes of the RC4 keystream. To do this, they first created a keystream byte distribution using many 128-bit RC4 initial keys. They then took these keys byte by byte and XORed them with a chosen plaintext candidate byte in order to get an induced distribution. From there, they just computed the most likely plaintext byte for each of the byte positions. Jacob mentioned, however, that this attack required the same plaintext to be encrypted under different keys each time. Jacob and his team found several ways to make this happen, such as by causing a client to continuously request access to a secure Web site via a session renegotiation or resumption. The second attack used a similar approach but involved known biases that exist within consecutive bytes in the entire RC4 keystream. Jacob pointed out that the full details of how this worked were in the paper. Other than the difference in the algorithm for the attack itself, this second attack requires the same plaintext to be encrypted with the same RC4 keystream. This precludes the need for any type of session renegotiation such as was required in the first attack. This attack is also not restricted to the first 220 bytes of the plaintext.

In terms of performance, the first attack showed an increase in percentage of plaintext recovered with an increase in the number of sessions used. In fact, Jacob and his team were able to achieve a plaintext recovery rate of 100% with a very large number of sessions. As for the second attack, the recovery rates were very high and scaled with the increased number of same plaintext copies. Despite these high recovery rates, both attacks required a vast amount of traffic to succeed and so were not practical. However, Jacob still suggested stopping the use of RC4 altogether as the most efficient way of preventing all of these attacks.

Someone asked whether these problems were caused by the TLS implementation or by TLS's interaction with RC4. Jacob said these problems were in fact due to how RC4 was implemented. The same person asked whether RC4 should still be used to protect credit card transactions online, as using RC4 is part of the standard for dealing with credit card information. Jacob said it depends. If you were using TLS 1.0 unpatched against a BEAST attack, for example, he would recommend just using RC4.

PCF: A Portable Circuit Format for Scalable Two-Party Secure Computation

Ben Kreuter, University of Virginia; Benjamin Mood, University of Oregon; Abhi Shelat, University of Virginia; Kevin Butler, University of Oregon

Ben first gave an overview of previous work on secure two-party computation. He pointed out that previous solutions to creating toolsets for two-party secure computation worked, but they suffered in their scalability. To fix this, Ben and his team developed not only a method to scale these secure computations, but also an entire library to do this called PCF.

Ben then went into several optimizations of previous work that make up PCF. One such example is that of reducing the storage size of circuits, particularly the storage of wire values, during runtime. Originally, a high-level language would be used to write the protocol and then compiled into a circuit. However, circuits can grow immensely during runtime depending on the protocol, such as with wire values. During runtime of a circuit, a table would be created for every wire, and then values would be put into the table entries. This creates a growing memory requirement that scales with the worst case to running time. Ben and his team used a simpler approach that overwrites wires when they are not needed using high-level information that the compiler can provide. Another improvement Ben discussed was that of PCF's flexibility with other languages. PCF can actually support any language for two-party computation. A developer, for example, could simply use standard C to program a protocol without adding any additional changes to the C language. As Ben put it, PCF can be thought of as simply writing and running a normal program.

Using this new tool, Ben and his team were able to handle billions of gates for a circuit. They were also able to reduce circuit file sizes and compile times by large orders of magnitude. Interestingly enough, Ben said that the actual bottleneck was in running the protocol itself.

Someone asked whether they ran into any counterexamples regarding the assumptions they made about the way that they were doing loops via backwards branches. Ben replied that they have not yet found any counterexamples, but they do have a backup plan if need be and a way to carry out the plan. Another person asked how they avoided information leakage if they are not evaluating the full depth of the circuit. Ben responded that only the branches in the forward direction can depend on private inputs. He added that for loops they rely on the user's ability to end the loop and thus do not terminate the loop if it happens to run infinitely, just like running a program.

Protecting and Understanding Binaries

Summarized by Xinyang Ge (xxg113@cse.psu.edu.ge)

Control Flow Integrity for COTS Binaries

Mingwei Zhang and R. Sekar, Stony Brook University

Awarded Best Paper!

Mingwei noted that control flow integrity (CFI) can mitigate attacks like buffer overflow attacks and return-oriented programming (ROP) that need to subvert the original control flow. However, previous CFI implementations rely on the compiler's help or debug information. Mingwei said that their work can apply CFI enforcement on stripped binaries.

The first challenge was to disassemble the binary. On architectures like x86, the instruction length is varied; there are "null" gaps between code which might be interpreted as instructions during disassembling. The authors combined linear disassembling with recursive disassembling to correctly identify gaps among code sections.

Binary instrumentation also requires transparency to existing code and maintaining the correctness of the original execution. To enforce control flow integrity over executable as well as all dynamically loaded libraries, they instrumented the Global Translation Table (GTT) that is used to map an indirect target with routing address in a different module. To keep the GTT updated, they modified the loader by adding 300 SLOC.

To evaluate the effectiveness of CFI enforcement, they proposed a metric called average indirect target reduction (AIR) that quantifies the fraction of eliminated indirect targets. They compared their techniques with others and showed the effectiveness of eliminating unnecessary indirect targets. To test the correctness of implementation, they applied their approach over more than 300 MB of binaries and the result was that none of them was broken during binary rewriting. Certain optimizations like branch prediction and adding address translation have been applied to the original implementation to reduce the overhead.

Ian Goldberg asked about how the gap is accurately identified. Mingwei answered they do not accurately identify the gap and it is possible the disassembler might mistakenly disassemble the gap. Since the gap would not be executed, it should be fine. Eric Bodden asked about self-loading libraries. Mingwei answered that all of the libraries should be translated in advance or CFI could not be enforced. And they haven't taken care of a self-loaded library so far.

Native x86 Decompilation Using Semantics—Preserving Structural Analysis and Iterative Control-Flow Structuring

Edward J. Schwartz, Carnegie Mellon University; JongHyup Lee, Korea National University of Transportation; Maverick Woo and David Brumley, Carnegie Mellon University

Edward first asked a question about whether researchers would like to read assembly or high-level language code like C. The answer is obvious: C code is much easier to understand than assembly code. And there are many existing techniques that require source code to do static analysis. Thus, their work focused on recovering the high-level abstractions from machine code.

The authors proposed two desired properties of decompilation: effective abstraction and correctness. To illustrate abstraction effectiveness, Edward showed two code examples doing the same thing, one using "goto" and the other using "while". To realize effective abstraction, they divided the decompiler, named Phoenix, into several components and recovered the control flow of the original program. A diagram illustrated how the decompiler works: (1) CFG recovery, (2) type recovery, (3) control flow structure, and (4) source code output. They captured the types by extracting the semantics of instructions. For instance, "movl (%eax), %ebx" reveals %eax is a pointer to type A while %ebx is of type A. With types, they further recover the control flow and generate source code. In order to preserve structuredness of source code, they apply iterative control flow structuring for source code generation. The aim is to minimize the use of "goto".

For evaluation, they showed an example decompilation of a short program and demonstrated the effective abstraction their decompiler can achieve. Then they launched some large-scale experiments with other decompilers (e.g., Hex-Rays, Boomerang) on GNU coreutils. They use two metrics to measure Phoenix: correctness and structuredness. The result turned out 50% of tested programs can be correctly executed and less goto's are used compared to other decompilers (details can be found in their paper).

Someone from UC Berkeley asked about whether their work focused on languages other than C. Edward answered currently their work focuses on C. Someone else asked about obfuscation or handwritten assembly. Edward said they are only looking at assembly directly from a compiler. Michael from UC Berkeley believed compiler optimization could change control flow. Edward said it is possible but if it represented the same logic, things should be fine. Scott Karlin (Princeton) suggested a further use case of detecting source code plagiarism. Finally, a researcher from Cisco asked whether they have tried multiple phases of compiling and decompiling using their tools. Unfortunately, the answer was no.

Strato: A Retargetable Framework for Low-Level Inlined-Reference Monitors

Bin Zeng and Gang Tan, Lehigh University; Úlfar Erlingsson, Google Inc.

Normally, attacks are launched by triggering existing bugs inside programs using user input. Previous countermeasures include data execution protection, address space layout randomization (e.g., PaX), and inlined reference monitors (IRM). An IRM is nothing but placing security checks inside programs. Most IRMs are implemented at a low level, which is difficult to reuse. Also, low-level instrumentations are restricted to a certain architecture and difficult to port. Thus, their work performs IRM rewriting at the Intermediate Representation (IR) level.

The challenge of doing IRM rewriting at IR is risky because the compiler is not reliable. For instance, the compiler might optimize the security checks out at the backend. So they intentionally add checks that are respected by the compiler and also verify that these checks are preserved after compilation is done. To illustrate how security checks are added to IR, Bin gave an example of IR code with checks added. In addition, they also did optimizations on the security checks including removing redundant checks.

To evaluate, they measured the performance on SPEC2k and portability by using same security checks on both x86 and x86-64. The average performance overhead was about 21%. For portability, the same instrumentation could work on both x86 and x86-64.

Someone asked about whether the security check is really ISA independent. Bin answered it actually depends on what the security check is. In fact, IR itself is not ISA independent. Ben Livshits (Microsoft Research) asked why the performance overhead is that high. Bin said intuitively this is related to the number of security checks placed, but they haven't measured what really incurs the overhead.

Invited Talk

Confessions of a "Recovering" Data Broker: Responsible Innovation in the Age of Big Data, Big Brother, and the Coming Skynet Terminators

Jim Adler, VP of Products, Metanautix

Jim Adler began his invited talk by introducing his company Metanautix. Metanautix is working on building a next generation big data management and analysis system. It has already built massive data analysis systems for many large enterprises such as Google, Facebook.

Based on his experience, Jim introduced the data supply chain. The huge amount of data from government, commercial, and self-reporting can generate huge value and are powerful for applications in transportation, marketing, etc. However, only few data collectors are regulated. Those unregulated uses can be easily abused by powerful people, and the hugeness and variety of data makes the world have less anonymity.

Through comparing EU rights and US torts, Jim asked, how do we unpack privacy and distinguish private from public? He further used place, player, and perils (3P) to characterize privacy issues. To describe the relationships among 3P, he concluded that player power gaps are proportional to secrecy and have an inverse relationship to trust. He further gave us an example of how his Felon predictor works (http://bloom.bg/1eMtnug) determining whether a person had committed a felony using other information in the database. He showed that the classifiers depend on policy as much as technology. Finally, he concluded that now government doesn't trust people but does trust machines.

Some people asked whether it is illegal to share private information on the market. Jim said it depends on what is privacy and what is public. Supermarkets usually do not share their customers' information with others. Some people were also curious about how to know which info is correct among huge data. Jim said through the data chain and huge data correlation, we have some mechanisms through which we can infer the valuable data. The world is shrinking in the information era, and we need to respect the data. Some people were worried about their privacy and asked whether we have choice to protect our privacy. Jim said that we need new policy now to deal with privacy protection. And we need better behaviors to protect our own privacy.

Current and Future Systems Security

Summarized by Sven Bugiel (bugiel@cs.uni-saarland.de)

On the Security of Picture Gesture Authentication

Ziming Zhao and Gail-Joon Ahn, Arizona State University and GFS Technology, Inc.; Jeong-Jin Seo, Arizona State University; Hongxin Hu, Delaware State University

Ziming Zhao presented his research on the security of picture gesture authentication (PGA) as deployed, for example, in the latest version of Microsoft's Windows 8 operating system. In PGA, users choose a background picture (from local storage) and perform gestures on this picture, such as tapping, drawing a circle, or drawing a line. The order, precision, and direction of those gestures then form the user password for authentication. To better understand the security of this new authentication mechanism, Ziming and his co-authors were first interested in better understanding the user-choice for background pictures and gestures. Using the results of this investigation, they devised and evaluated an automated attack framework to successfully break users' gesture passwords.

To investigate the users' choice of passwords (i.e., pictures and gestures), the authors conducted a user-study with two user-groups. The first group consisted of 56 computer science undergraduate students from Arizona State University, uniformly male, which used PGA for accessing class materials on the university Web site. The second group consisted of 762 participants recruited over public channels such as crowdsourcing, and their task was to emulate logging in to their online banking Web site using PGA. The study yielded that, from all picture categories, pictures depicting people are most commonly chosen since they are easier to remember, and that there is a strong relationship between the user's personality and his choice for his background picture. More importantly, the study showed that gestures are generally drawn around distinct points of interest, such as objects, shapes, or preeminent colors, and that the patterns for drawing gestures are very similar among different users.

Ziming and his co-authors applied these insights to design and implement an automated attack framework to break users' passwords. At the heart of their framework is a location-dependent gesture selection function that models and simulates the users' selection of gestures around and between points of interest. Evaluation of the attack framework based on the passwords collected from the user-study showed that the authors could successfully break between 24% (group 2) and 48% (group 1) of the passwords. This difference in success rate is explained by the lower security-sensitive context for the first group (access to class material), which resulted in simpler gestures (e.g., three times tapping a point of interest). Moreover, the evaluation showed that the attack success rate is noticeably higher for simple pictures with few points of interest and for portrait pictures with more predictable gestures. When tested as real-life online attacks on Windows 8 (i.e., only five attempts on guessing the gesture password) for passwords of the second group, the authors were still able to break 2.6% of the passwords.

The data sets of the user-study are available online at http://sefcom.asu.edu/pga/, and an example tool for measuring the gesture password strength is provided at https://honeyproject1.fulton.asu.edu/stmidx.

Chris Thompson (UC Berkeley) asked about the recall over time of gesture passwords and, further, if the two user groups are not too biased and participants of the first group are incentivized to chose weaker passwords. Ziming replied that they evaluated memorability of gesture password for the first group and the results are presented in the paper. The two groups were chosen on purpose in this configuration, and while users of group one did change their passwords to weaker ones, it is unclear why. Some feedback indicated that the weaker passwords were easier to use on smartphones. David Wagner asked whether the authors compared their real-life success rate of approximately 3% to the best attacks on text passwords. Ziming explained that they have not yet compared their results, but that the password space for picture gesture authentication is bigger than for text passwords, and this space could be further increased by allowing more gestures.

Explicating SDKs: Uncovering Assumptions Underlying Secure Authentication and Authorization

Rui Wang, Microsoft Research Redmond; Yuchen Zhou, University of Virginia; Shuo Chen and Shaz Qadeer, Microsoft Research Redmond; David Evans, University of Virginia; Yuri Gurevich, Microsoft Research Redmond

Yuchen Zhou presented his results in uncovering implicit assumptions by authors of authentication services' SDKs that can potentially compromise the security of applications that use those SDKs. As a result of this research, Yuchen and his co-authors were able to discover flaws in Facebook's authentication service and in the OAuth 2.0 specification.

Applications, today, are increasingly empowered by online services. One very prominent example is single sign-on (SSO) services offered by Facebook or Windows Live. To incorporate those services into their applications, developers are provided with SDKs and corresponding documentation on how to use the SDKs. Yuchen and his co-authors posed the question, whether the application is secure if the developer adheres to the SDK's documentation. He illustrated that this is not the case, by showing a demo video of an attack in which a malicious app is able to steal credentials retrieved from the Windows Live SSO service and use those credentials to impersonate itself as the legitimate user. Yuchen showed that such security issues can be traced back to implicit assumptions by the SDK developers, such as assumptions that are essential for the application's security properties and are not clearly stated in the SDK documentation, or that relate to how the SDK should be used.

To systematically discover such implicit assumptions in SDKs and their associated documentation, the authors of this paper built semantic models that capture both the logic of the SDK and the essential aspects of underlying runtime systems. To be able to consider all possible apps that can be built with an SDK, these models consider both the client and the service side. The semantic models, together with explicitly captured assumptions and security assertions (i.e., desired properties such authentication or authorization), form the input to a BOOGIE-based verifier. In an iterative process in which the model is refined or new assumptions are added, the final assumptions for this model are derived.

Applying this approach to explicate the three concrete examples of Facebook SSO PHP SDK, Windows 8 SDK for modern apps, and Windows Live connect SDK, Yuchen and his co-authors were able to uncover implicit assumptions that lead to a change of the Facebook SDK, a revision of the Windows Live SDK documentation, and an addendum to the OAuth 2.0 standard. Moreover, the authors conclude that due to these implicit assumptions, a majority of the tested apps—for example, Facebook's showcase apps—were vulnerable to attacks, and Yuchen illustrated this with concrete vulnerabilities for the Facebook SDK and Windows Live SDK.

Felix Lindner (Recurity Labs) asked about the efficiency of this approach versus a good Web-application pen tester. Yuchen replied this is a guided approach to better understand the system and find vulnerabilities. Penetration testing, on the other hand, is rather a black box testing to find vulnerabilities. Yuchen argued that their approach is more systematical but might help increase the efficiency of penetration testing. Someone asked whether the authors considered applying their approach more generally instead of only to SSO SDKs. Yuchen answered that their approach can definitively be generalized and applied to other models like payment, but they focused for now on SSO. Adrienne Porter Felt (Google) followed up on the different responses Yuchen received from the SDK providers and wondered whether updating an SDK documentation to include implicit assumptions is really enough. Yuchen replied that changing the SDK is definitively the best solution, because it is unclear whether developers adhere to the SDK documentation. But that is not always possible and thus documentation updates should be more strikingly propagated to developers to update their code.

Enabling Fine-Grained Permissions for Augmented Reality Applications with Recognizers

Suman Jana, The University of Texas at Austin; David Molnar and Alexander Moshchuk, Microsoft; Alan Dunn, The University of Texas at Austin; Benjamin Livshits, Helen J. Wang, and Eyal Ofek, Microsoft Research

Suman Jana presented a solution for a more fine-grained access control model for augmented reality (AR) applications, which simultaneously allows for a higher scalability of these applications. Suman first illustrated, based on different, very popular examples, such as the SoundWalk app or Google Glass, how AR applications abstractly operate: AR apps retrieve raw input from sensors such as the video camera, then apply object recognition algorithms (e.g., to detect hand gestures), and finally render the raw input augmented with virtual objects back to the screen. Currently, AR apps implement this pipeline by themselves and do not rely on operating system support.

Suman explained that this current status has two important drawbacks: first, since the applications retrieve raw, rich input, there is a high privacy risk. He illustrated this based on a face-recognition app that receives raw video camera streams and thus can also scan the background to discover, as an example, whiteboards full of confidential information. Second, the current AR application model does not allow two AR apps to run concurrently on the same hardware and hence does not scale.

The solution Suman presented is based on operating system support for augmented reality in the form of so-called "recognizers." A recognizer recognizes real-world objects from raw inputs (e.g., face or gesture recognition). AR applications can subscribe to recognizers and retrieve a stream of preprocessed data (e.g., the hand gestures performed or the recognized faces). Since applications do not retrieve raw input streams anymore, this enables a least-privilege access control for AR applications. To explain to the user which data an AR application receives, Suman and his co-authors introduced "privacy goggles," which previews to the user the filtered output; Suman provided different examples of privacy goggles in his presentation. Moreover, since the preprocessing of the recognizer can be offloaded and its output shared between different client apps, this allows for higher scalability of AR apps.

In their evaluation based on 87 Xbox applications, Suman and his co-authors discovered that 94% of the AR apps required access to the skeleton recognizer, used for tracking movements of a human body, and that only four recognizers (skeleton, person texture, voice command, and hand position) together cover about 90% of the tested applications. In addition, ten surveys with 50 participants each showed that 86% of the participants considered the recognizer output less privacy-sensitive. Suman presented that even with six apps sharing recognizers, more than 25 fps can be achieved for each app and he additionally showed the offloading of a heavyweight 3D modeling recognizer to an external graphic card. In future work, the authors want to investigate how to securely share the other steps of the processing pipeline among apps (e.g., rendering augmented output to screen) and how to securely support third-party recognizers.

Devdatta Akhawe (UC Berkeley) asked whether moving object recognition to the operating system level would result in a slower application development, since apps might require recognizers not yet available in the operating system and operating systems have slower update cycles. Devdatta wondered how many recognizers would be required for Xbox Kinect apps today, which were not available when the Xbox started shipping. Suman replied that they have no such statistics, but their evaluation shows that the bulk of the apps require only a few recognizers and that corner cases might be addressed in the future with a secure integration of third-party recognizers. Felix Lindner (Recurity Labs) wondered about the 14% of survey participants who were not able to use the privacy goggles despite the clearly unambiguous goggle preview. Suman mentioned that these users were rather boggled by the whole use-case and were unfamiliar with AR. Adrienne Porter Felt (Google) asked about barcode scanners as recognizers. Suman mentioned that this would be easily implementable and in fact they showed how to run a bar code scanner in a privacy-preserving manner in their S&P '13 paper, "A Scanner Darkly: Protecting User Privacy from Perceptual Applications."

Hardware and Embedded Security I

Summarized by Bhushan Jain (bpjain@cs.stonybrook.edu)

CacheAudit: A Tool for the Static Analysis of Cache Side Channels

Goran Doychev, IMDEA Software Institute; Dominik Feld, Saarland University; Boris Köpf and Laurent Mauborgne, IMDEA Software Institute; Jan Reineke, Saarland University

Boris Köpf started by discussing how caches improve performance by reducing memory accesses but also jeopardize security by leaking information about the latency for memory lookups. This leaked information can be used to recover secret keys from AES, DES, RSA, and ElGamal. He introduced the three types of cache attacks: timing based, where the attacker can determine the number of cache hits and misses from observing execution time; trace based, where the attacker can see the trace of cache hits and misses by monitoring power consumption; and access based, where the attacker shares a cache with the victim and can find information about the memory locations accessed by the victim. While some defenses against cache attacks are implemented in hardware, most of them are designed based on the interaction between the hardware and the software. These solutions depend on the cache specifics and the binary executing for security guarantee. CacheAudit helps such solutions to reason about the security guarantee using automatic static analysis of cache-side channels. It derives formal quantitative bounds on the information leaked to the attacker.

Boris then explained the theoretical foundations of CacheAudit. The goal is to compute a bound on the number of possible side-channel observations to give a quantitative security guarantee using program analysis. A binary program is represented by a state transition system where the cache is a part of the program semantics. The problem of computing the set of reachable states is not feasible. Abstract interpretation is a static analysis method where this set of reachable states is soundly over-approximated by using a set of abstract states that are mapped to actual states using a concretization function such that the abstract transition function always delivers a superset of the concrete transition function. Thus the size of superset of set of reachable states represents a bound on the number of reachable states at the end of program termination.

CacheAudit contains different abstract domains representing the states in stack, memory, flags, actual values, and cache hit or miss. It parses x86 code and generates a control flow graph that is traversed by the iterator to access all the possible states that can be reached. Boris did not go into much detail about cache abstract domain due to time constraints. The basic goal of cache abstract domain is to statically predict cache hits and misses. They analyzed the AES-128 implementation from the PolarSSL library. CacheAudit provides different bounds for different attacker models. Very few bits are leaked to timing based attacker and many bits are leaked to the trace based attacker. If the AES tables are preloaded, the bounds drop to 0 at the point where the table can be entirely in the cache. He directed the audience to the paper for many more results. The source code is publicly available.

Eric asked how to use the CacheAudit reports to distinguish between false positives and actual leakage. CacheAudit helps the security developer prove that the system is secure and allows him to make stronger security claims than before. Monitor the CacheAudit execution and analyze the location where the number of reachable cache states increases above one. Ben Livshits (Microsoft Research) asked about the loss in expressiveness if we go for zero leakage. Boris was not clear on the question. The chair suggested taking the discussion offline as the question and answer apparently needed some discussion.

Transparent ROP Exploit Mitigation Using Indirect Branch Tracing

Vasilis Pappas, Michalis Polychronakis, and Angelos D. Keromytis, Columbia University

Vasilis Pappas started with an overview of how code-level attacks and defense techniques have evolved. The code injection attacks were defended against by making data regions non-executable (DEP), so code injected there was neutralized. The attackers then started reusing the existing code (ROP). Address space randomization (ASLR) was introduced to defend against the code reuse attack. However, information leakage can break the ASLR protection very easily leaving the application vulnerable to code reuse, and ASLR is not used everywhere. Vasilis showed how return-oriented programming (ROP) works where the stack is set up so that a chain of gadgets are called using the return instruction without any code injection. The existing defenses either depend on the source code of the vulnerable application or disassembly of the binary for their analysis or defense against ROP attacks. This work is motivated to defend against the ROP attacks without any changes to the protected program.

kBouncer detects and prevents ROP code execution by monitoring executed indirect branches. It is transparent, compatible with code signing, self-modifying, JIT, etc. and causes 4% maximum overhead on artificial stress. They observed the runtime properties of ROP code like return instructions that do not follow a call site and a sequence chain of short code fragments through an indirect branch. kBouncer monitors at runtime the targets of any return instruction to ensure that return instruction targets valid call sites. However, an advanced ROP code can avoid illegal returns using jump-oriented programming or relying on call-preceded gadgets. To detect this advanced adversary, kBouncer inserts several short instruction sequences chained through indirect branches.

In order to improve efficiency, kBouncer leverages the Last Branch Record (LBR) from Intel architecture to store the last 16 executed branches. LBR is fully transparent to the running application, causes no overhead to record branches, doesn't require source code or debug symbols, and can be dynamically enabled for any program. kBouncer trades off the overhead and accuracy based on the frequency of reading the LBR cache. In order to further improve performance, kBouncer assumes that the ROP code will have to interact with the OS through system call. So, kBouncer checks for abnormal control transfer on system call entry.

kBouncer was implemented on Windows 7 x64. A gadget-chaining length of 16 does not produce false positives as gadget length never goes over 10. The performance overhead is minimal: 1% on average and 4% maximum. On detecting an attack, kBouncer pops up a window to the user that contains useful information to a developer. An attacker can either exploit the 16 entries on LBR or build a long call-preceded and non-return gadget and call it every time before calling a system call. The future work involves virtually extending the LBR size and raising the maximum gadget length to 40.

Kangjie Lu (Georgia Tech) asked whether kBouncer can defend against a two-fold attack in which the attacker first uses gadgets to set things up so as to call a virtual protect API but doesn't make the actual call, and then makes a bunch of random function calls to clear the information in LBR and call the virtual protect API as a second part of the attack. Vasilis agreed that this attack can bypass kBouncer but may be difficult to launch. Kangjie insisted that this attack was in fact very easy to launch. Eric Bodden asked if kBouncer could flag a legitimate program that uses more than 16 indirect branches in a row. kBouncer will not flag such programs if there are no instruction sequences where the targets are followed by another indirect branch. Simon (Georgia Tech) asked if LBR is saved by kBouncer on context switch as the OS, especially Windows XP, doesn't seem to be doing that. Vasilis agreed that kBouncer inherits the OS limitations and that with Windows 7, it saves only the last record. Simon suggested that in that case, the previous attack mentioned in the first question is even easier to launch as all the attacker has to do is force a context switch just before making the system call. Another person noted that when an application is attacked by ROP, the stack is sometimes ruined and the Windows API stops functioning; did they observe such behavior while reporting back the information in the popup window? Vasilis said that funny things may happen because of the stack being ruined but sometimes at least one thread is running well enough to post the message to the user.

Edward Schwartz (CMU) asked whether there is a problem if the ROP payload can write into executable memory and change the executable code. Vasilis said this will cause a code injection attack and will definitely be a problem but may be prevented by DEP.

FIE on Firmware: Finding Vulnerabilities in Embedded Systems Using Symbolic Execution

Drew Davidson, Benjamin Moench, Somesh Jha, and Thomas Ristenpart, University of Wisconsin—Madison

Drew Davidson presented their work on FIE, a symbolic execution tool that was built to find bugs such as memory safety violations and misuse of peripherals that occur in the firmware of embedded devices. FIE meets the firmware-specific challenges which violate the assumptions of traditional symbolic execution and can verify absence of bugs. Drew introduced an attack from the WOOT 2012 conference where a malicious smartphone is able to trick an Intuit GoPayment device that uses a MSP430 MCU into sending out the secret key by exploiting a buffer overflow in the firmware. Many other attacks on embedded devices are similar to these, and there is very little work on detecting vulnerabilities in firmware. The main idea of FIE is to transition successful source code analysis techniques for desktop like symbolic execution to be used on firmware.

KLEE is a popular and mature tool for symbolic execution. However, KLEE needs to be ported to the 16-bit environment to work with most firmware. Moreover, ported KLEE has terrible instruction coverage of less than 6%. Drew mentioned the three most prominent challenges of working with MSP430 code: (1) modeling architectural diversity based on how peripherals are accessed by I/O ports, and to a naive symbolic execution, accessing these ports looks like reading or writing to an uninitialized memory; (2) modeling peripheral semantics as the semantics of peripheral devices can be complicated and often breaks the assumptions of symbolic execution; (3) and modeling interrupt-driven programs as the control may transfer to the interrupt handler as long as the interrupt is enabled. FIE is highly customizable by using modules for chip layout spec, memory spec, and interrupt spec to overcome these challenges, respectively.

Chip layout spec is a flat text file written in a domain-specific specification language that allows the developer to represent memory size, memory region types, and available interrupts. The memory spec is a custom memory library that contains entries for memory/pin access consulted to get a symbolic value based on the constraints known to the developer. The interrupt spec library answers questions like whether the interrupts are enabled and what are their priorities. Once the interrupts are enabled, after every instruction, one path goes to interrupt handler and the other path of symbolic execution goes to the next instruction. FIE can also be used to verify the program by tracking all the possible states. FIE optimizes by avoiding execution further down a redundant state as all the successors to that state will also be redundant and will not reach new states. Drew referred the audience to the paper for more details on optimizations.

FIE was evaluated on Amazon EC2. They executed multiple versions of FIE using the 16-bit KLEE as baseline. FIE found 22 bugs in all and 21 of these were in the MSP430 USB protocol stack; the other bug was misuse of flash memory. Using the most optimization, FIE was able to reach 79.4% coverage as compared to 5.9% coverage for 16-bit KLEE, and more programs were verified when all the optimizations were applied. FIE verifies small programs well but large program verification is out of reach. In future, we may see dynamic testing, concolic execution, and static analysis techniques explored for firmwares.

Frank (University of Waterloo) asked whether it is worthwhile to add the ability to program the way external peripherals would act. Drew said they do that already where the developer can swap in custom libraries. External memory devices are mapped in via memory ports, and those ports can be encoded in the custom memory library. Indranil Banerjee (Qualcomm) asked whether it is a necessary condition that redundant states will always lead to redundant successors or just a heuristic. Drew answered that for a deterministic machine, that is pretty straightforward. In the case of a non-deterministic system, as the memory library is stateless, you don't have to deal with states and the symbolic memory addresses of any region that may contain non-determinism. Basically, all the non-determinism is captured in the state configuration. Jethro Beekman (UC Berkeley) asked how much work it would be to port FIE to non-MSP430 architectures. Drew replied that FIE took advantage of MSP430-specific features. While the basic techniques are sound, it would be a fair amount of work to port FIE to other architectures. Particularly, it would be easy to port to something with lower complexity.

Posters

Summarized by Jialong Zhang (jialong@cse.tamu.edu)

OS Lockdown on Industrial Control Systems with Process White List and Resource Access Control

Kuniyasu Suzaki, Toshiki Yagi, and Kazukuni Kobara, National Institute of Advanced Industrial Science and Technology; Yoshiaki Komoriya, Control System Security Center; Nobuko Inoue and Tomoyuki Kawade, SciencePark Corporation

Most of current industrial control systems are compromised through HMI (Human Machine Interface). HMI usually runs commodity OSes (e.g., Windows) which have many functions, applications, and resources that are not used for control systems but could be used by attackers. In this work, the authors proposed a new approach to reduce the attack interface by limiting processes and resources on the OS since it is enough to run limited applications on HMI instead of all kinds of applications. The proposed approach achieves this goal by building Processes White List (PWL) and Resource Access Control (RAC).

Anception: Hybrid Virtualization for Smartphone Applications

Earlence Fernandes, Ajit Aluri, Alex Crowell, and Atul Prakash, University of Michigan

The authors presented a hybrid virtualization system for untrusted apps to provide the isolation benefits of virtualization as well as a unified user experience. Specifically, untrusted apps must be bound to a lightweight VM, which essentially is an unprivileged set of processes executing in ring 1 of the x86 architecture. The proposed system only has 4700 LOC, with very minimal changes to existing Linux core.

To evaluate the overhead of the virtualization system, the authors ran it with 2D, 3D graphics benchmarks and the Sunspider app benchmark; results showed that the proposed system only caused slight slowdown in the apps (3.88% for 2D,3D graphics benchmark and 1.2% for the Sunspider app).

Entropic Return-Oriented Exploit Detection

Caleb Smith and Adam J. Aviv, Swarthmore College

In this work, the authors argued that current return-oriented programming (ROP) prevention systems are not practical due to the requirement for source code or debugging information, and current detection systems are often based on arbitrary or naive criteria (e.g., track-only instructions per return). From their study, they found entropy, jump/call/return address patterns and number of instructions between returns could be good features to detect a ROP exploit, and they propose a machine learning-based method to detect ROP exploits with those feature vectors.

HunterBee: An Advanced ZigBee Vulnerability Analysis System

Yuseok Jeon, Incheol Shin, Sinkyu Kim, Sungho Kim, and Jungtaek Seo, The Attached Institute of ETRI, Korea

ZigBee is a worldwide standard for wireless personal area networks, but there are lots of vulnerabilities in the ZigBee protocol. In this work, the authors presented an advanced vulnerability analysis system for ZigBee networks. Specifically, the whole system has an efficient network monitor model, simultaneous vulnerability assessment model, and security attack test model. The proposed system, HunterBee, can easily test known vulnerabilities and identify unknown vulnerabilities in different network scenarios through its simulation, emulation, and fuzzing functions.

To measure the performance of the HunterBee, the authors compared it with some commercial monitoring tools and vulnerability analysis tools. The results show that HunterBee is much more efficient and less interfering.

Protego: Practical Techniques to Obviate Setuid-to-Root Binaries

Bhushan Jain, Chia-Che Tsai, and Donald E. Porter, Stony Brook University

The setuid bit gives developers the flexibility to explore policies for new abstractions without changing the kernel. However, it also provides a chance for attackers to gain root privilege. Based on the authors' study, 89.5% of deployed systems have at least 28 setuid-to-root binaries. The authors proposed a method to identify precise security policy encoded by those trusted binaries and enforce it in the kernel. The proposed method uses LSM hooks in the kernel to check the system policy. As a result, among 40 historical exploits, the proposed method can successfully block all of them with low overhead (less than 6%).

A Method to Make Securing Your Information on Mobile Device

Yun-kyung Lee, Jae-deok Lim, and Jeong-nyeo Kim, Electronics and Telecommunications Research Institute, Korea

The authors proposed a new secure mobile device structure to protect user data in mobile devices. The new structure separates the original mobile device into two regions, the open region and the closed region. The open region is the general mobile OS region while the closed region is the secure and tiny real-time OS region. If a user wants to use a secure service on her mobile device, she first needs to register her app through a remote service server. Then if the user and applications authentication is successful on the mobile, it can create a session from the open region to the closed region to further utilize security functions (e.g., secure storage) in the closed region.

Android + Open WiFis = Broken SSL?

Sascha Fahl, Henning Perl, Marian Harbach, and Matthew Smith, Leibniz University Hannover

The authors conducted a study of 13,500 Android apps. They found that a large number of apps did not use SSL correctly and are vulnerable to man-in-the-middle (MITM) attacks. In this work, the authors study the MITM attacks against Android devices which use public WiFi and show how attackers can bypass apps that implement secure SSL certificate verification through open WiFis.

To evaluate the reality of such attacks, they conducted an Amazon Mechanical Turk study. Their results show that 73.4% of the participants were vulnerable to MITM attacks.

Classify but Verify: Breaking the Closed-World Assumption in Stylometric Authorship Attribution

Ariel Stolerman, Rebekah Overdorf, Sadia Afroz, and Rachel Greenstadt, Drexel University

Traditional stylometry research only focuses on closed world models, which limits to a set of known suspect authors. However, forensic analysts usually need open world models, in which some classes of authors may be unknown. In this work, the authors propose an abstaining classification approach that augments authorship classification with a verification step. The evaluation shows that the extended Sigma verification can have similar accuracy as traditional methods in closed-world problems.

An Automated System for Rapid and Secure Device Sanitization

Ralph LaBarge, Thomas A. Mazzuchi, and Shahram Sarkani, George Washington University

To reduce the amount of time and money spent on security, organizations often reuse compromised servers, switches, and routers. However, before redeploying these devices, they must be sanitized. In this work, the authors presented an automated system to rapidly sanitize servers, switches, and routers. The sanitation process will remove sensitive data and persistent malware and will restore the device to a trusted state. Their evaluation results showed that 14 Dell servers can be sanitized in about 70 minutes.

More Posters

Summarized by Rahul Pandita (rpandit@ncsu.edu)

The Effects of Developer-Specified Explanations for SmartPhone Permission Requests

Christopher Thompson, Serge Egelman, and David Wagner, University of California, Berkeley

The authors presented an interesting study done on a corpus of 4,395 free iPhone applications to determine the nature of the explanations provided by a developer to get runtime permissions from end users. These runtime permissions include access to locations, contacts, etc. In particular, they reported a very few developers provided meaningful explanations for using a runtime permissions (e.g., seven out of 200 applications pertaining to navigation specified the usage strings for access to location). Reporting a below average adoption of specifying usage strings, the authors proposed some interesting ways to better understand and tackle the problem in their future work; the first action item was a better study to understand adoption of usage strings across various developers.

When to Attack? Android UI State Inference as an Attack Building Block

Qi Alfred Chen, University of Michigan; Zhiyun Qian, NEC Labs; Sanae Rosen, Yuanyuan Zhou, and Z. Morley Mao, University of Michigan

The authors presented an interesting approach to assist security practitioners in determining the choice of an attack based on the user interface (UI) state of an Android device. They argued that most of the well known attacks can be executed under very specific circumstances. For instance, to carry out an attack to infer victims' keystrokes using the accelerometer data, the phone should be in a state where a victim is typing something on it. Knowledge of these circumstances greatly increase the odds of a successful attack.

The authors proposed to get this knowledge by leveraging the current UI state of an Android device under attack. In particular, they proposed a two-step methodology to infer the UI state. First, they detected the activity transition of windows-based events from the shared virtual memory. Second, they used the previously detected activity transition to infer specific activity.

A Survey of Fuzzy Hashing Algorithms for Malware Clustering

Jason Jones, Marc R. Eisenbarth, Michael Barr, and Alexandru G. Bardas, Arbor Networks

The folks from Arbor Networks had a practical work to present. They motivated the work by citing a large number of malware samples that a typical security company receives on daily basis. They further argued that the volume of the samples made it prohibitively time and resource consuming to perform manual inspection. This mandates some sort of automation to assist security practitioners. They proposed to automatically group the malware samples based on the class/families of malware.

The proposed grouping allows the security practitioners to look into the samples in an orderly/prioritized manner. In particular, they propose the use of fuzzy hashing (ssdeep, peHash, sdHash, and mvHash) algorithms on the incoming samples to perform this clustering or grouping. They claim that the proposed approach has produced encouraging preliminary results, and they are excited to perform more controlled evaluations in the future.

Towards Quantifying and Preventing the Leakage of Genomic Data Using Privacy-Enhancing Technologies

Erman Ayday, Jean Louis Raisaro, Mathias Humbert, and Jean-Pierre Hubaux, Ecole Polytechnique Fédérale de Lausanne (EPFL)

This work is an extension of the poster titled “Whole Genome Sequencing: Innovation, Dream, or Privacy Nightmare?”. Given the privacy implications of genome sequencing, the authors pointed out the need to secure the genome sequencing data. In particular, they proposed clear demarcation of authority (as in who uses such data when and how) and use of public key encryption to further secure these sequences.

Baton for Android: Key Agility Without a Centralized Certificate Infrastructure

David Barrera, Daniel McCarney, Jeremy Clark, and P.C. van Oorschot, Carleton University

The authors proposed an improvement to the existing security mechanism of application signing by a private key in the Android ecosystem. They argued that a developer with a compromised private key may need to change the key for application signing. Similarly a developer who has sold his/her app to another developer or company may want to change the key used for signing the app. Motivated by the previous arguments, the authors presented their system Baton, which is an extension of Android OS to assist the developer in changing keys used to sign applications, without relying on external certificate authority.

Whole Genome Sequencing: Innovation Dream or Privacy Nightmare?

Erman Ayday, Ecole Polytechnique Fédérale de Lausanne (EPFL); Emiliano De Cristofaro, Xerox PARC; Jean-Pierre Hubaux, Ecole Polytechnique Fédérale de Lausanne (EPFL); Gene Tsudik, University of California, Irvine

The authors educate people about the implications of leakage of genome sequences. They argued that a full genome sequence not only uniquely identifies an individual, it also provides additional information such as ethnic heritage, disease predispositions, etc. Such information can potentially lead to genetic discrimination. Additionally, this information can be leveraged by financial companies to deny or charge astronomical rates for health, mortgage, education loans, etc. Their work compiles a list of open research/ethical questions that need answering to address the privacy concerns of whole genome mapping.

Parameterized Trace Scaling

John Sonchack, University of Pennsylvania; Adam J. Aviv, Swarthmore College; and Jonathan M. Smith, University of Pennsylvania

The authors tackled the issues that are accompanied by data in a network experiments setting. First, live data is hard to come by. Second, usually the volume (size) of data itself is an issue. To alleviate these issues, the authors presented an approach that accepts a trace from a small network and uses this trace in a large scale simulation. In particular, to assist in simulation (for determining flow attributes) the system accepts user values for attribute distributions and correlation parameters. The authors presented encouraging preliminary results and look forward to integrating their tool with existing popularly used tools such as NS-3 and Mininet.

Security and Usability Perceptions of Android Password Patterns

Dane Fichter and Adam J. Aviv, Swarthmore College

The authors presented an interesting study to shed some light on the perception of a typical user towards the security in context of Android pattern passwords. In particular, they conducted a survey asking people to rank two patterns on the dimensions of usability and security. Among various attributes that authors considered for their survey, they report that the “length” of a pattern makes a lot of impact on security and usability of a pattern password.

While a smaller pattern is generally perceived to be more usable as it is easy to remember, people perceive a lengthy pattern as more secure. In the future, the authors want to apply their survey methodology to measure the perception of other password systems as well.

User-Intention–Based Android Malware Detection

Karim O. Elish, Danfeng (Daphne) Yao, and Barbara G. Ryder, Virginia Polytechnic Institute and State University; and Xuxian Jiang, North Carolina State University

The authors presented an interesting approach to classifying an unknown Android app as benign or malicious. They argued that if access to a sensitive resource (contact, location, etc.) in an Android system is not a direct outcome of a user-specified action (clicking of button, filling a text-box) in the application UI, then most likely the application is malicious. They have a preliminary empirical evaluation to validate their claim and are working hard on extending this work to get further insights.

Fingerprint Me If You Can: Towards Effective Protection Against Browser Fingerprinting.

Henning Perl, Sascha Fahl, and Matthew Smith, Distributed Computing & Security Group, Germany

The authors motivated this work by highlighting the increasingly sophisticated techniques employed by Web sites to track visitors, thus creating a potential privacy threat. In particular, the authors proposed to generalize, obfuscate, overwrite, and remove specific values in browser properties to safeguard a user against the browser fingerprinting.

Hardware and Embedded Security II

Summarized by Muhammad Naveed (naveed2@illinois.edu)

Sancus: Low-Cost Trustworthy Extensible Networked Devices with a Zero-Software Trusted Computing Base

Job Noorman, Pieter Agten, Wilfried Daniels, Raoul Strackx, Anthony Van Herrewege, Christophe Huygens, Bart Preneel, Ingrid Verbauwhede, and Frank Piessens, KU Leuven

Job started his talk by explaining the Carna botnet port scanning map which exploited routers with a default password. The authors' goal was to design and implement a low-cost and extensible security architecture that provides strong isolation of software modules, secure communication and attestation, and thwarting attackers using a zero-software trusted computing base.

Software modules are isolated using program-counter based memory access control which enables variable access rights, isolation of data and protection against code misuse. Isolation can be enabled and disabled using the commands protect and unprotect, respectively.

A trusted third party provides key management. Symmetric cryptography is used because public key crypto is computationally expensive for embedded devices. The trusted party is needed only for initialization and after successful initialization modules can be deployed without any intervention from the third party.

The node and software provider compute a symmetric key. This key, which can only be calculated by the node and the software provider, is used for secure communication by producing a MAC of the data. Remote attestation in Sancus is achieved using secure communication.

To enable intermodule function calls, a caller module is deployed with the MAC of the callee module's identity using the caller's key, and whenever it is called the caller checks the MAC and only then allows the function call.

Someone asked how the key revocation works in Sancus. Job replied that they have no way to detect the compromised keys but are proposing what needs to be done after learning that a key has been compromised. All keys except the node key can be replaced. If a node key is leaked, the software provider needs to change its identity. Someone asked whether the modules are verified before the isolation is enabled. Job answered no, because this is not necessary due to Sancus' key management scheme.

Securing Computer Hardware Using 3D Integrated Circuit (IC) Technology and Split Manufacturing for Obfuscation

Frank Imeson, Ariq Emtenan, Siddharth Garg, and Mahesh V. Tripunitara, University of Waterloo

Awarded Best Student Paper!

Frank started by stating that the integrated circuit (IC) is the physical realization of digital logic. The manufacturing process of an IC starts with modeling using a hardware description language (HDL) which is converted into an optimized digital circuit, using a netlist, which is then used to fabricate the IC.

The threat model for the proposed approach assumes that an insider in the external foundry (IC manufacturer) is malicious and can maliciously change the IC die. This enables several attacks (e.g., privilege escalation attack, leakage of private information, etc.). A successful attack in the context of this talk was to be able to uniquely identify and change the output of at least one gate.

Frank explained how a malicious gate could be inserted to a full-adder netlist affecting the output of the adder. The proposed solution to prevent modification of the circuit by the manufacturer is circuit obfuscation, which hides some of the wiring in the netlist.

3D IC technology has two or more tiers connected via bond points, and the proposed approach exploits this use of multiple tiers. The hidden part of the circuit is fabricated in-house on one tier while the obfuscated circuit is shipped to the manufacturer to be fabricated on another tier. An attacker generally needs to identify one or more gates in the circuit, and this process is impeded by the missing wire connections thus confusing the attacker with at least k indistinguishable choices for each gate in the circuit. The authors defined k to be their notion of security (k-security). A greedy algorithm for trading cost for security was presented, and the experiments showed that the proposed greedy algorithm performed much better than randomized selection.

The proposed approach makes the attack difficult because now the attacker needs to attack k (k is a parameter determining security of the circuit) gates as opposed to attacking a single gate, which means a higher chance of detection.

An MSR researcher asked about the security for different batches. Frank replied that in that case the process needs to be repeated for every batch. Cynthia Sturton(UC Berkeley) asked what would be the best use case for this technology. Frank replied that a particular subset of the circuit can be secured to lower the cost, but the cost for security needs to be paid. Kevin Fu (University of Michigan) asked about the actual physical cost. Frank noted that there is a cost to make the circuit 3D, and they may not be able to use all the surface area of the chip. Someone wondered how the authors evaluated the cost of the attack and the outcome of the successful products since multiple chips are made out of a single wafer, and the attacker can distribute possible attack points on multiple wafers. Frank answered that if the attacker can mount four types of attacks for an identical circuit, then their approach needs four times the amount of security for single attack. The last question was about the chance of attack, whether it's possible to attack an individual circuit or the whole batch. Frank answered that the hardware attack needs a lot of upfront work, the attacker has only one chance to attack, and a successful attack affects the whole batch. Attacks on a single circuit are not possible.

KI-Mon: A Hardware-Assisted Event-Triggered Monitoring Platform for Mutable Kernel Object

Hojoon Lee, Korea Advanced Institute of Science and Technology (KAIST); HyunGon Moon, Seoul National University; DaeHee Jang and Kihwan Kim, Korea Advanced Institute of Science and Technology (KAIST); Jihoon Lee and Yunheung Paek, Seoul National University; Brent ByungHoon Kang, Korea Advanced Institute of Science and Technology (KAIST)

A kernel integrity detection mechanism that lies in the kernel space can be attacked by rootkits in the same kernel space. Therefore, kernel rootkit detection needs to be executed in a separate execution environment—for example, on top of a hypervisor or as a separate hardware unit. Existing kernel rootkit detection schemes are limited to blind kernel static region write detection and can't monitor mutable kernel objects. Value verification is required to stop malicious function calls, while semantic verification is required to thwart the loading of malicious LKMs.

Hojoon Lee presented the KI-Mon platform, which provides a hardware-assisted mechanism to build rootkit detection rules. For value verification it uses hardware-assisted memory whitelisting and for semantic verification it uses an event-triggered callback mechanism. KI-Mon consists of a KI-Mon processor that inspects the integrity of events, a Value Table Management Unit (VTMU) that snoops the host system bus and generates a HAW-Event that is then passed to the KI-Mon processor, and a DMA module that monitors the system memory and sends memory contents to the KI-Mon processor for integrity inspection.

KI-Mon provides an API for programmability, and this API consists of the following four layers: raw data layer, data structure layer, semantic layer, and monitor layer.

A snapshot-only monitor vs KI-Mon comparison was presented, showing that KI-Mon detected an LKM insertion event that the snapshot-only monitor missed. The snapshot-only monitor is computationally expensive while KI-Mon has zero overhead during idle times, and even during the LKM insertion event it consumes orders of magnitude fewer cycles than the snapshot-only monitor in idle mode.

Someone asked how the memory snapshots were analyzed. Lee replied that the data acquisition engine of KI-Mon reconstructs kernel data structures from raw memory snapshots. Someone asked how KI-Mon dealt with page table manipulation attacks. Lee said that in the current prototype, KI-Mon performs virtual to physical address translation using the host's page tables and doesn't have any integrity monitoring scheme for the host page tables. But they are currently investigating the possibility of such attacks. The last question concerned how many monitoring rules KI-Mon supports. The answer was only two rules.

Invited Talk

Summarized by Stephen Crane (sjcrane@uci.edu)

Security Team 2.0

Chris Evans, Google Chrome Security Team

In this interesting look into the security team responsible for one of the world's most popular browsers, Evans presented what he believes sets their team apart from the majority of traditional security teams, including their innovative use of vulnerability rewards programs. Evans, who founded the Chrome security team in 2009, began with a brief history of Chrome's security, dating back to their acquisition of primary sandboxing technology for the browser. He continued by summarizing a few of the major accomplishments of his security team, some of which are enumerated online at http://www.chromium.org/Home/chromium-security/brag-sheet.

Evans focused on contrasting traditional, somewhat inefficient security teams with some of the novel approaches taken by the Google security teams, and the Chrome team especially. The first of these comparisons centered around Google's policy of "fix it yourself." In the traditional model, the security team operates primarily as consultants and sends issues and bugs back to the development team to fix. In contrast Google's security team frequently codes, and often lands security fixes themselves. They have found this to promote both knowledge of the code base in the team, as well as respect and cooperation from the development teams. Developers see the security team as an asset and ally, rather than a nuisance loading them down with additional work. Closely coupled with fixing bugs themselves, the Chrome security team also takes the position that everything is in scope for security. If it affects the end user, it needs to be secure. Because of this, the entire stack, and especially third-party plugins are targets for the team.

Evans further related how the Chrome security team actually goes the extra mile, rather than just talking about how security is important. The team aggressively makes sure to backport security fixes, even private, internally found bugs, since a malicious party has probably found the same bug. He also mentioned their efforts at fuzzing to find bugs not only in their own browser, but also in third-party plugins, and even occasionally other browsers entirely. In contrast to much of the traditional thinking, Evans believes that security is not a zero-sum game, and the Chrome security team is encouraged to research not only their own browser, but also to collaborate to secure software from other vendors.

Another factor that Evans credited as crucial to the success of his security team is their streamlined organizational process. Rather than interacting with end users and researchers only through approved public relations channels, the security team converses directly via the Chromium bug tracker. This transparency and interaction is critical not only to understanding and fixing bugs quickly, but also to building user trust. After security bugs are fixed, their complete history is made public, which is both a great research resource for the security community and an encouraging record of the diligence of the security of Chrome. Finally, this public interaction helps to foster community involvement, which Evans has found essential to their work.

In the last section of his talk, Evans discussed the Google vulnerability rewards programs. Google gives out monetary rewards to researchers who find security bugs in both the Chromium browser and many Google Web services. Evans and his team have found that this program has worked wonderfully, and recommends a similar program as an industry best practice to any company which is in the front-line of defense against malware. To date, Google has paid out over $2 million, split almost equally between the Chromium and Web services programs. Google started by creating a rewards program for reporting vulnerabilities in Chromium. Building on the success of this program, Google then started a similar rewards program for many of their Web applications. Evans related that this required significant discussion between the security and legal arms of the company, but ultimately has turned out extremely well.

In evaluating these rewards programs, Evans had nothing but praise for their success and effectiveness. While there was initially concern over the quality of Web application bug reports, they found this was not a problem. Evans highly recommended instituting a vulnerability rewards program, since they and others have been extremely satisfied with the program. However, Evans advised that any security team wishing to start such a program must be well prepared and overstaffed to accommodate the initial influx of bug reports. He also advised starting small and encouraging bug reports for beta or unreleased versions of software.

Alan Sherman (University of Maryland) posed the question of how security teams can act proactively starting with the beginning of project design. Evans replied that it is critical that security have a seat at the table during discussions of new features or projects. This gives the security team the opportunity to evaluate risk and, if needed, assign staff during the design phase to make sure security is factored in from the beginning. Chris Watzac (Georgetown University) then praised Google's "good neighbor policy" in securing third-party code, but asked if they ever tired of fixing Flash. Evans, smiling, replied that they are certainly not tired of helping to secure Flash, but that they do try to automate much of the fuzzing and vulnerability testing. Following up on fuzzing, Eric Eide (University of Utah) asked about Google's fuzz-testing infrastructure, known as clusterfuzz. Evans explained that they use a more than 2,000-core cluster to fuzz at scale. He recommended to start simple with fuzzing, then progressively tune the fuzzer to be more clever and understand more of the application domain, as needed.

Finally, Zhiyun Qian (NEC Labs) inquired how many security engineers are at Google, and what the ratio of security to the rest of the product team was. Evans was unsure on exact numbers, but indicated that company-wide there were many hundreds of security engineers. He also noted that while the industry recommendation is to have 1% of engineers focused on security, Google has a significantly higher percentage than this. Finally, he stated that the Chrome security team has about 20 engineers, along with many domain experts they consult with who are not officially on the team.

Mobile Security II

Summarized by Ziming Zhao (zzhao30@asu.edu)

WHYPER: Towards Automating Risk Assessment of Mobile Applications

Rahul Pandita, Xusheng Xiao, Wei Yang, William Enck, and Tao Xie, North Carolina State University

Rahul started by noting that recent work has developed various techniques to determine what mobile applications do. Such approaches include manual inspection adopted by Apple where Apple employees read an application's description and determine whether the resource used by this application is appropriate. Google also does such checking by using Bouncer which is a static and dynamic malware analysis tool and permission system; however, there was no work talking about what mobile users expect mobile applications to do. The authors took the first step to answer this question by focusing on permission systems in current mobile phones. In particular, they wanted to understand whether an application's description provides any indication why the application needs certain permissions. Rahul then presented their high-level approach using natural language processing techniques to determine why an application needs certain permissions.

Rahul presented their framework, WHYPER, which combines why and permission. WHYPER uses natural language processing instead of keyword-based searching, because keyword-based searching cannot deal with keywords with confounding meanings, and it lacks the ability to make semantic inference. He gave an interesting example: the description "Share with your friends via emails, sms" does not mention the need for the permission of "reading contacts," however it does infer the usage of contact information.

Rahul explained some natural language processing preliminaries they planned to use, such as parts of speech tagging, phrase and clause parsing, typed dependencies, and named entity recognition. The WHYPER framework includes a pre-processor to parse application descriptions and an intermediate-representation generator to transform pre-processed sentences to first-order logic representations (FOL). Then a semantic engine takes FOL and generates semantic graphs of permissions. As the result, Rahul showed that WHYPER could achieve 97.9% accuracy for read contacts permission and 96.8% for read calendar permission.

Someone from Microsoft Research said the descriptions of applications sometimes do not really reflect their use of permissions, so she asked whether the proposed approach is really reliable. Rahul replied that WHYPER is more like a first line of defense and cannot achieve comprehensive and complete malware analysis and detection. Someone from MIT asked how a malware developer can take advantage of the low accuracy of natural language processing in this system for their purpose. Rahul replied the most easy way is to insert more dummy descriptions.

Effective Inter-Component Communication Mapping in Android: An Essential Step Towards Holistic Security Analysis

Damien Octeau and Patrick McDaniel, Pennsylvania State University; Somesh Jha, University of Wisconsin—Madison; Alexandre Bartel, University of Luxembourg; Eric Bodden, Technische Universität Darmstadt; Jacques Klein and Yves Le Traon, University of Luxembourg

Damien started his presentation by explaining that intents (which include the descriptions of action, category, and data) are the vehicles for inter-component communication in Android and can be either explicit or implicit. In order to receive intents, a program needs to specify an intent filter in its manifest file. Even though there is work to expose and analyze, the interfaces provided by components to interact, existing approaches are ad hoc and imprecise. Damien and his co-authors proposed recasting inter-component communication analysis to infer the locations and substance of all inter- and intra-application communication available.

Damien defined the goals of their work which was to infer specifications for each ICC source and sink in the targeted applications. These specifications detail the type, form, and data associated with the communication. He further explained the inferred specifications could be used to find ICC vulnerabilities, identify attacks on ICC vulnerabilities and analyze inter component information flow. In particular, their framework outputs the following information for given applications: a list of entry points in an application that might be called by the application itself or other applications; a list of exit points in an application that this application may share intents with other components; a list of links between the source and sink of intent communications.

To this end, Damien presented the main part of their analysis approach which is based on an IDE framework (interprocedural distributive environment). An IDE framework solves a class of interprocedural data flow analysis problems. In these problems, an environment contains information at each program point. For each program idiom, environment transformers are defined and modify the environment according to semantics. Damien used lambda calculus to explain environment transformers.

Damien presented their tool which is called Epicc (Efficient and Precise ICC). As shown in his presentation, the precision for their tool to analyze explicit ICC is around 98% and 88% for implicit ICC. Their tool is available at http://siis.cse.psu.edu/epicc/

The session chair Shuo Chen asked whether their approach supports other channels besides intents. Damien replied their technique is actually applicable to other channels such as content provider, and they were looking into extending their work to these modules. Someone from Kansas State asked how they handled alias analysis. Damien replied that they used an alias analysis tool from Soot which is flow sensitive and context insensitive.

Jekyll on iOS: When Benign Apps Become Evil

Tielei Wang, Kangjie Lu, Long Lu, Simon Chung, and Wenke Lee, Georgia Institute of Technology

Tielei started his presentation by showing the cover of the novel Strange Case of Dr Jekyll and Mr Hyde. Tielei used the title "Jekyll on iOS" to indicate the possibility of using dual personalities on iOS to do something evil. Tielei said that Apple was confident about the security of iOS and denied the requests for antivirus apps. He explained that Apple is so confident because in addition to standard security techniques, such as data execution prevention, address space layout randomization, sandboxing, encrypted file system, and privilege isolation, Apple also adopts intense mandatory application review and mandatory code signing. In the history of iOS, only a handful of malicious apps have been discovered.

Tielei first explained what app review and code signing can do. In the best case, the app review guarantees that all existing execution paths do not contain malicious intents, and code signing guarantees the static integrity of the application. On the other hand, app review and code signing cannot check the dynamic integrity of an application, which means new execution paths could be introduced by subverting the original control flow.

Therefore, the control flows someone experiences using on iOS could be different from the control flow checked by Apple.

In order to utilize this insight, Tielei and his co-author deliberately created a vulnerable application and published it in the App Store. Since there is no immediate malicious behavior of this app, it passes Apple's review. After it is installed in end users' iOS, the authors tried to exploit the vulnerabilities in this app and introduce new execution paths. To achieve this, they needed to bypass several security mechanisms. They used an information leakage vulnerability to bypass address space layout randomization and used return-oriented programming and return-to-LibC to bypass data execution prevention. He also showed source code examples of the intended vulnerabilities.

Tielei demonstrated their results by showing how to post tweets without user permission in iOS. He further explained that Jekyll is like a new attack surface on iOS which could freely invoke private APIs that are not allowed to be used by third-party applications. Jekyll could also be used to attack other applications and even the iOS kernel and drivers. Tielei said they removed their application from App Store immediately after it passed Apple's review and only ran experiments on their own devices. They informed Apple of this vulnerability before the USENIX Security conference.

Tielei also discussed the possibility of detecting Jekyll. He said detecting such an attack surface is very difficult in the review phase but it is quite possible with runtime mitigations by using a fine-grained permission model, sandbox profile, and even runtime monitoring.

Shuo Chen asked what was Apple's response when the authors disclosed their results. Tielei said they just appreciated their work. Someone from Penn State asked whether their demo malware required the same permission as a normal app to post tweets. Tielei said he was not sure at that moment and he would double check to answer that question. Someone from Japan asked whether the change of control flow required restarting the app. Tielei answered no.

Large Scale Systems Security III

Summarized by Matthias Wählisch (m.waehlisch@fu-berlin.de)

Measuring the Practical Impact of DNSSEC Deployment

Wilson Lian, University of California, San Diego; Eric Rescorla, RTFM, Inc.; Hovav Shacham and Stefan Savage, University of California, San Diego

Wilson talked about practical impact of DNSSEC deployment. He started by asking whether security is better with DNSSEC than without it. The authors provided insights into this provocative question. Deploying a new security protocol may introduce unintended side effects on legacy devices. Based on a measurement study using ~530,000 globally distributed DNS clients, the authors found that ~1% of clients are not able to resolve DNSSEC-signed resources.

DNS is a major building block of the current Internet. In its original version, it is vulnerable to tampering, i.e., an attacker is able to corrupt the DNS answer, and a DNS client is not able to verify the integrity of the reply. This leads to an invalid name to address mapping. Quite recently DNSSEC has been deployed, which allows DNS zone administrators to sign DNS records. DNS requesters (in particular recursive DNS servers) can thus verify the authenticity of DNS data. In their study, the authors measured the end-to-end DNSSEC behavior for three different types of domains: legacy domains, correctly secured domains, and misconfigured DNSSEC domains. For the latter, 25 domains simulated DNSSEC tampering.

After some brief background, Wilson highlighted that their large-scale measurement study was challenging as it required forcing end users to request specific names. An easy approach would be a patch-based extension of Web browsers. However, they have no relation with Mozilla or Chrome, for example, and such an approach might raise ethical concerns. Common research testbeds such as PlanetLab also fail since node distribution is not representative. Likewise, asking friends or colleagues to install a measurement client introduces a bias. To overcome these problems the authors decided to use a Cash-4Traffic Web advertising concept and integrated JavaScript into advertisement banners. The requested ad images were stored on servers that are named with domains from the sample set.

The results of the study solely rely on server-side access logs to prevent inconsistent analysis which results from error-prone browser behavior. Wilson reported an increased failure rate in browsers with the number of subsequently loaded images. From a methodological point of view, this implies a random load of ad banner images.

The data set gathered by the authors includes ~530,000 clients using 35,010 unique resolvers located in 6446 unique ASes. The main reason for the errors lies in oversized DNS packets, which require TCP fallback or UDP path MTU prediction. These mechanisms do not work properly in the DNS regime. The erroneous resolvers were highly localized in Asia.

Avishai Wool (Tel Aviv University) commented on the different TCP/UDP behavior. He assumed that this was most likely due to firewalls and middleware. Several operators assume usage of TCP-based DNS traffic only for zone transfers, where UDP is open for client to resolver communication. Wilson agreed with this.

Frank Imeson (University of Waterloo) asked how many attacks against DNS really exist. Wilson replied that they did not study this measure but it depends on the services reachable via domain. Banks probably experience more attacks than do blogs, for example. Their measurement data is especially important for those domains that need DNSSEC protection.

The session chair, Micah Sherr (Georgetown University), was curious whether the authors had taken into account sampling biases due to ad blockers. Wilson admitted that such artifacts were possible but they did not look into that.

Jethro Beekman (UC Berkeley) was interested in a more detailed location of one of the measurement points (ISP X). Wilson replied that ISP X was in the Philippines. The rationale behind Jethro's question was to find a reason for the large portion of failures the authors observed. It might be related to censorship in China. Wilson made clear that most of the locations for ISP X were not in China.

Haixin Duan (Tsinghua University) asked whether the authors simulated an attacker which injects another DNS answer for the resolver such that the resolver gets two duplicated DNSSEC replies. The authors did not try this particular attack.

This talk immediately initiated a little discussion on NANOG: http://mailman.nanog.org/pipermail/nanog/2013-August/060400.html.

ExecScent: Mining for New C&C Domains in Live Networks with Adaptive Control Protocol Templates

Terry Nelms, Damballa, Inc. and Georgia Institute of Technology; Roberto Perdisci, University of Georgia and Georgia Institute of Technology; Mustaque Ahamad, Georgia Institute of Technology and New York University, Abu Dhabi

Terry began by noting that even large enterprise networks need to cope with the problem of not having control over all end hosts (e.g., guests), which may carry malware. In contrast to host-based protection, network solutions externally monitor network traffic. Using a prototypical deployment of ExecSent in real networks, Terry and his co-authors were able to find more than 25,000 new infected machines.

There are three common approaches to perform network-based malware detection: anomaly-based, domain-based, and URL-regex, Terry explained. However, these methods do not work properly in large enterprise networks. He illustrated the rationale behind ExecSent by two observations they made. First, C&C protocols change infrequently due to code-reuse of malware. Second, HTTP is the most prominent choice for a C&C application layer protocol. Terry described the ExecScent architecture, which mainly gets two inputs, background network traffic and malware traffic traces, and generates Adaptive Control Protocol Templates. A template represents the structure of the malware under investigation, is self-tuning, and looks at the entire HTTP request and not only at the attacker URL. Template matching is performed by measuring and comparing similarity and specificity of occurrences.

Terry highlighted that ExecScent was deployed for two weeks in three real networks (two universities and one large US financial institution). Their ground truth was represented by (1) commercial C&C blacklists, (2) top 1 million Alexa sites for whitelisting, and (3) analysis by professional threat experts for domains that are not part of either (1) or (2). Looking on the C&C domains they found, half of the domains were new in the university network, and all domains detected in the financial network had not been discovered before. In total, they identified 65 brand new C&C domains. They used the gathered information in six ISP networks for one week and found 25,586 new potential malware infections.

Finally, Terry critically discussed the limitations of ExecScent. Even though ExecScent depends on malware traces and labeled domains, they need only one malware sample to create an Adaptive Control Protocol Template for a complete malware family. An attacker may bypass ExecScent by implementing a new protocol or changing IP addresses. However, this increases the complexity for malicious users. Blending into background traffic or injecting noise into the protocol is also very effective because ExecScent works adaptively and cares more about the components in the structure that the malware request has in common than the differences.

Richard Johnson (NCAR) asked whether they integrate ExecScent into existing security information management (SIM) systems. Terry replied that this technology is part of a product, which has at least the capability to do this. Tudor Dumitra (University of Maryland) was interested in how much of the C&C traffic was done over HTTPS. Terry explained that they found only a very small portion of HTTPS traffic in the malware feed they got and thus ignored it. A colleague from Carleton University, Canada, asked for clarification on the deployment in the six ISP networks. Terry answered they did this in collaboration with a partner who deployed the information on DNS sensors.

ZMap: Fast Internet-Wide Scanning and Its Security Applications

Zakir Durumeric, Eric Wustrow, and J. Alex Halderman, University of Michigan

Zakir reported on a new Internet scanning application they developed, ZMap. With this open-source tool a user is able to scan the entire IPv4 address space for one port from one commodity machine in less than 45 minutes. It achieves a 98% coverage of live hosts.

He compared ZMap with existing scanners such as Nmap and highlighted that ZMap was explicitly designed to scan the entire Internet. It operates fully asynchronously and does not require per-connection states. They never wait for timeouts. ZMap does not track individual hosts and does not initiate retransmits but always sends a predefined number of probes. Zakir emphasized that ZMap scans as fast as the network allows. To allow for the very large number of parallel connections, they optimized the network stack and directly generate Ethernet frames.

Zakir explained how they overcame the problem of excessive states. Using random permutation and iteration over multiplicative groups, they do not need to store the complete address scanning plan but only information about the next host to contact. They validate responses without local per-target state by encoding secrets into mutable fields of probe packets that will be recognizable as responses from particular scanned addresses.

Among other performance results, Zakir presented a comparison with Nmap. They found that ZMap scans 1300 times faster than Nmap and identifies more results. This is due to the short timeout in Nmap's aggressive mode. Furthermore, he discussed application scenarios for high speed scanning, as well as aspects of good Internet citizenship while scanning. In the future, they want to perform a 10 GbE scanning with ZMap, and extend it to IPv6 capabilities. They also will think about scanning exclusion standards such that an Internet host can ask not to be scanned.

Someone wondered whether the authors observed decreased responses during repeated scanning. Zakir replied that they are not aware that they have been blacklisted. Vern Paxson (UC Berkeley/ICSI) pointed to work by Bellovin and colleagues about fast IPv6 scanning. Richard Johnson (NCAR) was interested in measured effects of ZMap on IDSes (e.g., Argus). Zakir said that they did not perform a systematic analysis and can only provide some anecdotal evidence. Nikita Borisov (University of Illinois) asked about the disadvantages of not keeping states. Can Acar (Qualcomm) gave a historical note that some tools which implemented asynchronous scanning already existed ten years ago. In connection to Nikita's question, Vern Paxson asked whether Zakir was able to validate responses. Zakir said yes because they embed data into sequence numbers, ports, etc., and stressed that they have enough bits for this. Someone asked how they can encode metadata in source port and IP address and yet receive the packet back. Zakir explained they applied a complete /24 network for source addresses, which provided variety. They also used a dedicated scanning host, which retrieves packets via pcap, and thus allowed them to exploit the entire port range.

Web Security

Summarized by Yuchen Zhou (yz8ra@virginia.edu)

Eradicating DNS Rebinding with the Extended Same-Origin Policy

Martin Johns and Sebastian Lekies, SAP Research; Ben Stock, Friedrich-Alexander-Universität Erlangen-Nürnberg

Ben Stock provided some basics about Web applications and the same origin policy (SOP). Then he gave a brief review on what a DNS rebinding attack is and how it can be used to exfiltrate information from an intranet behind a firewall. Basically the attacker lures the victim to visit a particular Web site where the attacker has control over content as well as the domain's DNS server. A short lived (TTL=1) query is issued to the client so subsequent requests would require another DNS query, to which the malicious DNS server responds with the IP of a machine from the intranet. SOP fails because it only checks domain, port and protocol, but not the actual server's identity. Ben then introduced several real-world DNS-rebinding attacks from as early as 1996. However, browsers have since been fixed by pinning the DNS to a fixed IP per execution so that the well-known rebinding attack has been mitigated.

Attackers can carry out this attack again using the new HTML5 offline caching feature. Due to the recent introduction of this feature and APIs for Web sites to specify the duration of the cached contents, the attacker is able to plant an attack script in the victim's machine for a longer duration. Although DNS pinning makes sure the IP is fixed to the attacker's server during this run, it loses its effect when the user closes the browser. However, the attacker's script doesn't go away after the browser restarts, thus opening up the attack vector. Ben indicated that the underlying problem to this type of attack is that current same origin policy doesn't involve the application server. To mitigate this, Ben showed their new protection scheme—extended SOP that is supposed to be deployed at both the client and the server side. On the server side, it is supposed to provide an extra header indicating the origin of this resource. On the client side, instead of checking only protocol, domain and port, the browser also checks for this extra origin header provided by the server.

Someone asked whether eSOP prevented against a malicious party modifying the origin header, and Ben said that is a different threat model than what they considered. They assume the attacker cannot arbitrarily modify network traffic. So his scheme doesn't offer any integrity check on the header. Shuo Chen (MSR) noted that if an intranet needs authentication, the user would be prompted to enter credentials (by which time he/she would realize that an attack is going on). Ben said yes, the rebinding attack would be less successful in that situation.

Revolver: An Automated Approach to the Detection of Evasive Web-Based Malware

Alexandros Kapravelos and Yan Shoshitaishvili, University of California, Santa Barbara; Marco Cova, University of Birmingham; Christopher Kruegel and Giovanni Vigna, University of California, Santa Barbara

Alex Kapravelos began by saying that drive-by downloads are still one of the more popular attacks. He then provided an example of drive-by downloading JavaScript. Although this behavior can be easily detected by honeypots, he showed that scripts can hide their behavior by some clever manipulation, such as tricking the honeypot into loading an unavailable activeXObject. Unless the load fails, the attack won't happen. Because the honeypot's browser isn't a real browser and always allows such loading to be successful, this malicious behavior won't be detected by the honeypot.

Alex presented Revolver, the system they developed to infer program evolution history to help detect such evasion techniques.

The tool itself does not offer vulnerability detection capabilities, so they require an oracle to tell whether a given script is malicious or not. In this work they used wepawet developed by UCSB (themselves) to serve this role. Revolver takes in JavaScript and parses it into AST, which is then broken further down to node sequences. These serve as the features for the classifier. Before actually comparing two scripts to see whether they are similar, they need to first eliminate irrelevant candidates to speed up the process. They did various optimizations such as approximate nearest neighbor matching (to reduce the candidate pool) and combining small scripts into larger ones to prevent attackers from evading Revolver by breaking the attack scripts down to smaller pieces. They named four classes of results from Revolver and the oracle: "injection," which corresponds to scripts previously detected as benign but later evolved to malicious; "evasion," meaning scripts previously malicious but now appearing benign; "data-dependency," or scripts that are always benign; and "evolution," scripts that are always malicious. They have evaluated their system on 6 million Web pages of which 265k are malicious, containing a total of 20 million benign scripts and 186,000 malicious scripts. The analysis revealed several interesting evasion techniques, and he explained two of them in the slides.

Somebody asked what Revolver can do if the attackers switch to metamorphic JavaScript attacks. Alex said that currently Revolver cannot detect complicated metamorphism, but the fact that Revolver is not reporting any evolution could be suspicious and this would prompt the researcher to investigate the script more carefully.

Language-Based Defenses Against Untrusted Browser Origins

Karthikeyan Bhargavan and Antoine Delignat-Lavaud, INRIA Paris-Rocquencourt; Sergio Maffeis, Imperial College London

Karthikeyan Bhargavan began by stating that previous works on Web component isolation only focused on protecting the hosts against potentially malicious embedded scripts. Few people have looked at how to protect embedded scripts from malicious hosts. Karthikeyan gave several examples including single sign-on buttons/scripts, password completion bookmarklets, and client-side encryption scripts. All of the above components need to be protected from a malicious host. To write a completely isolated, outside world–independent module is not impossible but extremely hard. The developer needs to use closures, forbid inline scripts, use crypto mechanisms, make their script self-contained, and maybe more just to hide a secret. Therefore, a component programming framework that offers strong isolation and protection from hosts is needed.

Their approach is to first limit the component's JavaScript usage to a subset called DJS that is amenable to analysis. Any communication between trusted modules and scripts can be done using their DJCL library, which implements popular encryption schemes. They also implemented a tool called DJS2PV that takes in a DJS-compatible script and translates it into a pi-calculus model that can be used by popular COTS verifiers such as ProVerif to reason about certain security properties automatically. The speaker alluded to proof and type systems and referred listeners to the paper for more details. After talking about the mechanisms, the presenter showed two implementations using their toolchain: a Facebook login module and a password bookmarklet.

Somebody asked how this approach differs from previous JS subset works. The speaker claimed that their approach focuses on defending embedded components from the outside world, therefore needs different mechanisms than SES or Caja, etc.

Invited Talk

Summarized by Sven Bugiel (bugiel@cs.uni-saarland.de)

Building Securable Infrastructure: Open-Source Private Clouds

Pravir Chandra, CTO Security Architect, Bloomberg

Pravir Chandra shared his experiences with the Bloomberg Clustered Private Cloud (BCPC) project for building a private cloud with open-source software. In particular, he focused on his insights for security challenges and opportunities that such an infrastructure provides. Pravir mentioned that he and his team publicly share those experiences and provide a set of recipes used to build the Bloomberg private cloud. These recipes are available online at www.github.com/bloomberg/chef-bcpc.

Pravir explained their motivation for building a private cloud. To better illustrate the scale of the infrastructure, Pravir gave some numbers: about 3500 employees of Bloomberg are actively coding; Bloomberg Professional Services has about 300,000 subscribers; and the infrastructure connects about 200 countries. This large-scale infrastructure requires, as Pravir explained, a centralized management for assigning resources in a highly flexible manner while simultaneously avoiding single points of failures and cascading failures. A cloud infrastructure covers these requirements. However, because of too-high latencies and the sensitivity of the data, a public cloud provider was not an option, and Pravir and his team opted for building the open-source Bloomberg Clustered Private Cloud.

Next, Pravir introduced the BCPC architecture, which builds on the open-source OpenStack cloud operating system, but incorporates several other open-source solutions such as Apache Web server. Specifically worth mentioning is the Ceph file-system, which allows for physical fault-tolerance (e.g., specifying in which rack a file copy has to be stored) and thus enables a reliable distributed object store. The storage traffic is routed through a dedicated network without a single point of failure. Similarly, the message bus and state database required by OpenStack are made Byzantine fault-tolerant through a distributed MySQL database and by using clustered RabbitMQ as message bus.

Pravir highlighted the particular benefits of BCPC for development and production efficiency. Developers are now able to quickly spawn virtual machines to experiment with novel ideas in a safe and sandboxed fashion. This is in particular beneficial for setting up test environments accessible to internal and external users. Moreover, automation of setting up virtual machines can automatically resolve all dependencies of test environments and thus improve deployment of new code. Production efficiency benefits primarily from the improved scalability of BCPC, which allows new virtual machines to be deployed within seconds to cover peak traffic and which provides commodity storage solutions for improved storage performance and deployment flexibility. Moreover, production efficiency is improved through standardized package management systems that ensure easily repeatable deployments and through providing developers with access to logs and monitoring data of their virtual machines.

Subsequently, Pravir elaborated in more detail on the security benefits of their BCPC. First, thanks to the high flexibility of their cloud, virtual machines can be easily dedicated to one specific task and thus allow an efficient isolation and sandboxing of tasks. Pravir illustrated this benefit at the example of more effectively rolling out security patches to virtual machines. Moreover, it offers machine impermanence—that is, if a virtual machine were compromised, it could simply be torn down and restored from a known, trusted state. Other security layers implemented in their cloud infrastructure are out-of band audits and surveillance (introspection) from a control-plane, compartmentalization enforced by the underlying hypervisor, automated background patching of software, runtime integrity checks (e.g., running a tripwire-like service from the hypervisor level), and network access control lists derived from declarative specifications. Further ideas, which might be added in the future, include hypervisor-based integrity checking of VMs at the process level, utilizing trusted computing devices to envelope VMs from the hypervisor, and further protection mechanisms against VM break-out vulnerabilities.

An extensive Q&A session followed. Felix Lindner (Recurity Labs) asked whether they looked at the PrivateCore solution. Pravir replied that they did and it looks interesting, but has not been deployed in BCPC. Someone asked whether Pravir and his team considered the integrity of the message queue and state database separately. Pravir answered that they consider them as part of the hypervisor level and hence trusted computing base. Another question targeted the attacker model and which threats should specifically be prevented in BCPC. Pravir explained that they consider both external and insider attackers, and the main protection mechanisms are automation to remove errors and compartmentalization to isolate errors and security breaches.

Another question considered data exfiltration attacks. Pravir replied that they did not consider data exfiltration at the service level, but protect against exfiltration at the hypervisor level. Another person followed up on the idea of using trusted computing devices and what kind of devices Pravir had in mind. The answer was that they considered HSMs (hardware security modules) and in fact designed their own PCIe-based HSM.

Someone asked about how they implement VM introspection when the memory is randomized with ASLR. Pravir explained that if only the hypervisor memory is randomized then VM introspection is feasible. Another question mentioned that the NSA is currently getting rid of a majority of their system administrators and whether Pravir and his team have any insight into this through using extensive automation. Pravir explained that it was not their goal to reduce the number of administrators but the number of people with access to the infrastructure level and that this reduces the insider threat profile. Following up on the topic of automation, someone else asked whether the high amount of automation increases the BCPC attack surface. Pravir explained that more software means more complexity, but the front-end of their infrastructure, which is exposed to attacks, is very small. Another questioner enquired about the heterogeneity of the software used in their BCPC and how they deal with that. Pravir replied that they provide a centralized managed software repository, that they sensitize people to use this managed software, and that they provide recipes on how to integrate software into their cloud. The last questioner asked where the firewall rules (i.e., network access controls) are enforced. Pravir answered that it is a mix of software firewalls in the hypervisor and hardware firewalls between tenants. Usually enforcement is north-south, but if two tenants are very close, enforcement can also be configured to be east-west.

Invited Talk

Summarized by Bin Zeng (zeb209@Lehigh.EDU)

Tracking the Casino Computer Wars: Who's Winning—the Casino or the Cheat?

Richard Marcus, Casino Cheating and Fraud Consultant; author of *American Roulette, The Great Casino Heist, Dirty Poker, World's Greatest Gambling Scams,* and *Identity Theft, Incorporated*

Security is much broader than hacking into computers. In this entertaining talk, No. 1 casino cheat and yet computer-illiterate Richard Marcus started with a demonstration of what he's dubbed "The Savannah" move. This involves betting a low denomination chip such as $5 chip on top of a maximum denomination chip, say, $5,000 chip. This seemingly easy but extremely-difficult-to-pull-off move turns surveillance system into his allies. Then Marcus moved on to the increasing use of computer technologies in gambling—RFID technology, roulette computers, laser scanners, and so on—by both cheats and casinos. Marcus discussed hacking into online gambling and casino security systems before concluding with a question into how Australian cheats hacked into a video surveillance system and won$32 million. The talk was extremely entertaining and engaging, and the non-casino-literate audience was surprisingly responsive.

Attacks

Summarized by Frank Imeson (fcimeson@gmail.com)

Take This Personally: Pollution Attacks on Personalized Services

Xinyu Xing, Wei Meng, and Dan Doozan, Georgia Institute of Technology; Alex C. Snoeren, University of California, San Diego; Nick Feamster and Wenke Lee, Georgia Institute of Technology

Xinyu's talk described how Web sites like Google, YouTube, and Amazon personalize the content on a per user basis to make the user experience better and or allow these services to make more money; however Xinyu poses and answers the question as to whether personalization is actually secure. He answers this by presenting their exploit for YouTube that allows them to insert two videos into your recommended video list with a high probability.

YouTube keeps track of navigation statistics for which videos users navigate from and to: if most users who watch video A then watch video B, when you watch video A, YouTube will recommend that you watch video B. YouTube's list has more than one recommendation per video, but it also uses two slots to recommend videos that are related to your viewing history. The proposed attack is executed when you visit a third-party Web site that secretly loads videos into your YouTube history, and these maliciously inserted videos now cause YouTube to recommend a set of videos that the adversary wants you to view. This is called a pollution attack. Of course, the adversary needs a set of videos to insert into your history, and these videos need to have a high navigation rate to the videos that the adversary wants you to view. One option is for the adversary to create a new set of videos and have all his Facebook friends watch it, then immediately navigate to adversarial videos. Unfortunately for Xinyu, he does not have many Facebook friends, so he instead used cross-site scripting to automate the process. The results varied but if you as a YouTube user have a small amount of YouTube history, then you are more vulnerable to this attack. Xinyu concluded by stating that he has shown that this type of attack is effective and that it can be mounted on other personalized services.

The session chair, Thorsten Holz (Ruhr-Universität Bochum), asked whether the content provider can detect a pollution attack, either by detecting the action of polluting a user's profile or by analysis of the user's profile. Xinyu said that if the video link is put in an iframe, it may play in its entirety without the user knowing, and thus YouTube cannot simply filter non-played videos to avoid this attack. As for detecting whether the user's profile has been tampered with, it is difficult to detect maliciously inserted videos. Fish (Ruoyu Wang—U C Santa Barbara) appreciated this approach and asked for an example of using this for malware or another type of attack. Xinyu replied that this attack is completely orthogonal to malware. Frank Imeson (University of Waterloo) said that it is in the best interests of the services to offer protection against these attacks, but can this be done on the server side or does it have to be executed on the client side? Xinyu said that server-side detection is difficult, and the client's browser has more potential to implement a solution.

Steal This Movie: Automatically Bypassing DRM Protection in Streaming Media Services

Ruoyu Wang, University of California, Santa Barbara, and Tsinghua University; Yan Shoshitaishvili, Christopher Kruegel, and Giovanni Vigna, University of California, Santa Barbara

Ruoyu Wang, aka Fish, presented a method for getting around the Digital Rights Management (DRM) of streaming Internet video which is implemented specifically to protect against piracy. The video streaming from server to client is encrypted, the client side software decrypts and decodes the video so that it can be displayed onto your computer screen. Fish et al. exploit the fact that decrypted/decoded (raw) video is handled by the client software. They developed software that monitors the video streaming client software to grab the raw video from the buffer and reconstruct the entire video in perfect quality that can now be sold on the black market.

Of course the process of how the system works is not trivial and in fact the system takes up to 24 hours of setup time before the movie can be pirated in real time. The general idea behind the software is to monitor the running binary and its memory with a custom Pin tool, the setup process allows the Pin software to locate the loop responsible for decoding the video which in turn helps train a classifier to distinguish which memory in the buffer is raw video. After the setup process, the raw video can be efficiently captured from the buffer and reassembled into the pirated copy. Fish et al. have followed responsible disclosure with Microsoft, Spotify, Adobe, Amazon, Netflix, and Hulu. Microsoft, Spotify, and Adobe provided positive feedback. Fish also presented a few mitigation techniques, but the one that got the biggest laugh was when he claimed the best way to get them to stop was for the companies to sue him.

Someone asked how much of the buffer is needed for the classifier to properly identify raw video. Fish stated that for a reliable judgment to be made, about 800 kilobytes should be fed into the classifier. Simon Chung (Georgia Tech) asked whether the decrypt/decode loops are the bottlenecks of the client software or whether it can be obfuscated. Fish informed him that although they have not modified the video software it does seem bottlenecked, but they can obfuscate the buffer order or use virtual machine protections with not much overhead. Someone else asked what would be the best counter measures. Fish joked that other then suing them, buffer obfuscation and/or spreading out the decryption task in many loops to confuse their program would work for this version.

Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer

Roel Verdult, Baris Ege, Radboud University Nijmegen; Flavio D. Garcia, University of Birmingham

This talk was arguably the most controversial talk at USENIX Security '13, as such the attendance was expected to be high and so the organizers massaged the timeline to make this the last talk of the day. And this seemed to work as the ballroom was full. Roel Verdult presented a subset of his work, starting out by explaining that he is not allowed to present the paper but he can at least present the title of the paper, which received a good laugh. Due to a recent injunction by the High Court of London this talk could not contain the core technical content of the unpublished paper, so he had to stick to the slides and could not answer any questions.

Roel started with an overview of vehicle immobilizers, which are used as electronic anti-theft devices. They prevent hot-wiring, use an RFID transponder, are mandatory in a number of countries, and use digital signatures for mutual validation, but as the title suggests the security protocol is not sufficient to prevent the guys from the University Nijemgen from breaking it. Since he cannot present his most recent work he instead talked about a related attack on the similar technology of Hitag2, which has been shown to have a weak cryptographic algorithm that uses no randomness, does not enforce atomic operations for the secret key update, has not been properly used by car manufactures, and only has a key length of 48-bits. He described how to attack Hitag2 in an academic setting and explained that the steps involved are not very practical for your everyday criminal. He then went on to describe a similar technology, Megamos, which has similar weaknesses and attack practicalities, but is still deployed in cars.

Next Roel reviewed the responsible disclosure aspect of publishing attacks, which consists of alerting the manufacturers of the device well before the attack would be made public. Roel and his colleagues at Nijmegen University followed these best practices, but the High Court impeded their ability to share their work due to the nature of this attack. Roel then walked us through different techniques for reverse engineering these systems: observing input/output behavior and decompiling the firmware and chip slicing, which allows the adversary to study the system at a microscopic scale and look for vulnerabilities. He then talked about how his attack uses the Tango Programmer and is on the Magamous Crypto algorithm which is partially publicly available, but that is the extent of what he is allowed to share about the attack.

Roel reviewed a subset of proposed mitigation techniques from cryptography literature: using a publicly accepted encryption standard like AES, using modern updating schemes like Patch Tuesday (or something that could work for automobiles), and he also points out a company, Atmel, that took an open design approach to allow for public evaluation. He concluded by stating that the automotive industry concentrates a lot of their efforts on safety, but not enough on security. They are still using proprietary and outdated algorithms. He ended the presentation by presenting a hash of the paper that they cannot publish as a historical claim, which received a good laugh and much applause.

Since we could not ask questions of the presenter, the session chair, Thorsten Holz (Ruhr-Universität Bochum) thanked him for giving us the best presentation that he could and informed us that he was going immediately get his computer grinding away on that hash. He thanked the program chair, Sam King, all of the attendees, reminded us that USENIX has an open access publishing model, "your papers are yours, the copyright is yours," and asked us for our support by becoming a member and signing up for next year's USENIX Security '14 held in San Diego, to be chaired by Kevin Fu, University of Michigan.