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

Conference Reports

WOOT '13: 7th USENIX Workshop on Offensive Technologies

Washington, D.C.
August 13, 2013

Summarized by Judson Dressler and Rik Farrow

Network Attacks I

Summarized by Rik Farrow (rik@usenix.org)

Truncating TLS Connections to Violate Beliefs in Web Applications

Ben Smyth and Alfredo Pironti, INRIA Paris-Rocquencourt

Alfredo Pironti began by explaining that they had found they could exploit Web application logic by disrupting TLS by closing the connection. For example, a wire transfer to "Charlie%27s_Angels" could become one for "Charlie" if the packet were fragmented after "Charlie" and the connection closed before the second packet was sent. Pironti said that the solution was as simple as correctly designing the application protocol so that transfers only occur if the length of the payload is correct and the graceful closing of the TLS connection.

Pironti described several attacks, starting with the Helios electronic voting system used at the Catholic University of Louvain and Princeton University for electing student governments. Helios has the feature that participants can vote multiple times, but only their final vote will count. Smythe and Pironti had noticed that the sign-out request is exactly 701 bytes, so an iptables rule can be used to drop these packets, leaving the user logged in. Although they notified the authors of Helios in 2012, the Web application remains unpatched. Pironti suggested that by making transactions atomic and notifying users about the current state of the system (vote cast, signed out), the system could be defended against this attack.

In his second example, Pironti demonstrated a flaw in Microsoft Live's authentication logic. The attacker needs the ability to control the network (drop packets) and access to the computer after an authenticated user believes she has logged out. For example, to read Hotmail, the user gets redirected to account.live.com for authentication, but then reads her email from a different server. When logging out, it is this server that sends a "signed-out" message, so by dropping the traffic to the authentication server to prevent the sign-off message from reaching that server, the user appears to have signed out, but her session is still active. Pironti used a video to demonstrate this attack. Pironti provided a similar example when a user has two open sessions with Google, and the attacker can reset the connection before the image representing the closing of the session is sent. Pironti's presentation was crystal clear, and there were no questions.

FireDrill: Interactive DNS Rebinding

Yunxing Dai and Ryan Resig, University of Michigan

Yunxing Dai explained how DNS rebinding attacks work. The goal of a rebinding attack is to get a browser to violate the same origin policy (SOP) through replacing a correct DNS response with one that has an address pointing to an internal server that should not be accessible by the attacker. The attacker, who must lure the victim to his Web site, uses his special DNS server to send two different responses to "attacker.com". The first has a time-to-live (TTL) of one second, so it expires immediately. The victim's browser begins to download a Web page from attacker.com, and that Web page includes a request to include another resource from attacker.com. On the second DNS request for attacker.com, the special DNS server replies with an address internal to the victim's site. Once that content has been loaded into the victim's browser, JavaScript from the attacker's initial Web page can access that content and communicate it to the attacker.

Dai said that the browser vendors responded to this attack by "pinning" DNS responses within the browser. Dai pointed out that browsers have a limited amount of memory dedicated to caching pinned DNS information, so the attacker can include requests to many subdomains, causing the pinned entries to be flushed from the cache.

Rik Farrow asked why not use the subdomains to inject the phony DNS reply? Dai replied that you run into the same SOP problem. A person from the University of Paris pointed out that there was a way this would still work: by starting with a subdomain within an iframe, the parent could disconnect that subframe and then use the parent domain to respond with the internal server address.

Subverting BIND's SRTT Algorithm Derandomizing NS Selection

Roee Hay, IBM; Jonathan Kalechstein, Technion—Israel Institute of Technology; Gabi Nakibly, National EW Research & Simulation Center, Israel

Roee Hay explained both past and new attacks for DNS cache poisoning. Dan Kaminski made these attacks famous several years ago, showing how easy it was to inject fact responses into a DNS server's cache. The response to those attacks in BIND was to randomize portions of the recursive server requests to make responses harder to spoof. BIND developers used the destination IP, the TXID, and the source port to provide a greater degree of randomness than had traditionally been used.

Hay pointed out that there have been other studies on the randomness of the TXID and source port portion of the nonce in requests, and they focused on the name server (destination IP) portion of the nonce. By derandomizing the selection of the name

server, existing attacks become more efficient. Also, an attacker who can see requests, but not responses because of non-symmetric routes, will be able to execute man-in-the-middle attacks.

BIND uses the smoothed round trip time (SRTT) algorithm to choose which of possibly many authoritative name servers will be queried by a recursive name server. SRTT is used to pick the name server with the smallest latency. Their attack uses an authoritative name server under the attacker's control to trick the resolver under attack to query many other non-open name servers for answers multiple times, which results in the lowering of the SRTT value of the chosen name server, making it more likely to be chosen. ISC, the maintainer of BIND, has acknowledged the issue, but considers it non-critical; a patch will be part of the normal release cycle.

WOOT: Mobile Attacks

Summarized by Judson Dressler (jd25@rice.edu)

Bluetooth: With Low Energy Comes Low Security

Mike Ryan, iSEC Partners

Bluetooth Smart is different from traditional Bluetooth at the PHY and link layer but reuses high level protocols. It is prevalent in high-end smartphones, fitness devices, and, more recently, in medical devices. At the PHY layer, it uses Gaussian frequency-shift keying over 40 channels in the 2.4 GHz frequency range. Of the 40 channels, three are for advertising and 37 for data with channel hopping.

Mike Ryan implemented a Bluetooth Low Energy (aka Bluetooth Smart) monitor on a CC2400 Ubertooth platform, matched the hopping pattern using Fermat's Little Theorem, and then handed the data off to the microcontroller to packetize. Once in packets, the stream is dumped to pcap. The AES encryption is strong, so the key exchange protocol was attacked using a Wireshark module to extract the keys from the initial communication setup. The defense for this attack is the use of a real key exchange protocol.

The question and answer session revealed that the sniffable range of Bluetooth Low Energy is approximately 10 meters and that traditional Bluetooth is more secure due to its use of more information in the hopping function.

Breaking Cell Phone Authentication: Vulnerabilities in AKA, IMS, and Android

Jethro Beekman and Christopher Thompson

Jethro Beekman began with an overview of the three phases of a phone's authentication and key agreement. These are identity, challenge, and response. With telephony over IP, there are no session keys used. Based on this fact, the authors demonstrated that they could create a false base station ("man in the middle") attack, a malware attack, or an imposter attack. In order to do the false base station attack, the attacker would need the read_phone_state permission, which happens to be available to over one-third of all of the apps tested. This attack severely impacted T-Mobile's WiFi calling as it allowed for man in the middle eavesdropping attacks.

T-Mobile updated their networks with an over-the-air patch in March of 2013. The false base station attack could be thwarted by using Digest AKAv1 or update to AKAv2 and use TLS. In either case, confidentiality and integrity controls must be used. Users should also limit access to the internal API.

The question and answer session revealed that in their fix, T-Mobile included their root certification for authentication purposes. Also, a comment revealed that read_phone_state permission is now the default for all apps written for Android 4 or later.

Cloning Credit Cards: A Combined Pre-Play and Downgrade Attack on EMV Contactless

Michael Roland, Josef Langer, NFC Research Lab Hagenberg, University of Applied Sciences Upper Austria

Contactless credit cards are based on ISO 14443 using 13.56 MHz inductive coupling technology. Michael discussed one type of EMV contactless credit card, Mastercard Paypass. Paypass in EMV mode uses secure chip and pin design but also allows for backwards compatibility with the magnetic stripe mode. The EMV card contains static card data (account number, expiration date, etc.), issuer's digital signature, public keys of card and issue as well as the private key of the card for signing.

When downgrading to mag-stripe mode, there is no offline authentication. Using this knowledge, the authors' goal was to skim and duplicate credit cards. Mag-stripe is supported at all terminals, and the terminal cannot check the integrity of static data on the card, therefore this data is easily skimmed. The difficult part of cloning the card was the CVC3 number, which is a function of an unpredictable number, transaction counter, and secret card data.

The preplay attack relies on determining the unpredictable number. The unpredictable number is a 4-byte field but is limited due to BCD coding, defined by the issuer, and stored on the card. EMV card backwards compatibility actually reduces the possible CVC3 numbers to approximately 1000 values. Using the skimmed data, these possible values are precomputed and stored on the cloned card. The authors attempted three cloned cards by two issuers at three terminals; each time payments were approved.

Improvements to current methods would be to notify issuers of downgrade cases and to increase the number of CVC3 digits. Mastercard has acknowledged vulnerabilities and stated that their protocols and rules provide countermeasures; it is up to the card issuers to implement them.

Network Attacks II

Summarized by Judson Dressler (jd25@rice.edu)

Leveraging Honest Users: Steath Command-and-Control of Botnets

Diogo Mónica and Carlos Ribeiro, INESC-ID/IST

Botnets are continuing to evolve and botmasters are coming up with new strategies to avoid takedown/detection. With this as motivation, Diogo Mónica wanted to explore new directions that future command and control infrastructure might take. He started with the assumptions of an already infected population and that this population has a trust anchor and can receive commands. He then began describing a possible command and control network with no active participation, where bots listen for commands, and all commands are signed by the botmaster and pushed out to all bots. The advantages of no C&C network is no infiltration or size estimation can be accomplished, but it makes it difficult for the botmaster to disseminate commands and retrieve information.

To alleviate this issue, Diogo described an expendable layer of hosts that have no knowledge of the botmaster and which can do all of the "heavy lifting" of seeking out and contacting the infected bots. In order to accomplish this, he devised a browser-based attack that directs the "honest" users to send out the commands for him. This reduces the botmaster's vulnerability of detection, the speed of dissemination increases with more browsers helping, and the commands are received but never acknowledged making it impossible to determine the size of the network itself. To retrieve information, Diogo described two different methods. For spam bots, the information can just be sent in the spam itself, therefore all the botmaster has to do is receive the spam. For more sensitive information, the botmaster creates a Web site with a public-private key pair, then searches the IP space for one infected bot. From here, that bot takes over to find the next one, creating an encrypted chain of infected bot IP addresses.

From an IP Address to a Street Address: Using Wireless Signals to Locate a Target

Craig A. Shue, Worcester Polytechnic Institute; Nathanael Paul, University of Tennessee and Oak Ridge National Laboratory; Curtis R. Taylor, Worcester Polytechnic Institute

Online criminals must be apprehended and current IP location services are not precise (only to 690 meters) or fast enough for law enforcement. Without the use of special law enforcement practices, Curtis Taylor presented a method for fast and precise localization of an IP address that is universally applicable and minimally invasive. Their method consists of three components: a target, a signaler, and an observer. The signaler sends packets to the target at pre-established sizes and times using out of window TCP packets, avoiding the problems of encryption and NATs while not alerting the target.

Curtis performed two experiments—in an apartment building and in a residential neighborhood—using no directional antennas or signal strength. He was able to narrow the location down to three houses in the residential area and the correct unit in the apartment building. The countermeasures to this geolocation method are using a hardwire connection or a proxy server, router configuration to obfuscate packet size, or including packet size in an anomaly detection device.

This presentation demonstrated that users are not anonymous and anyone can do this tracking, which raises privacy/legality concerns.

Looking Inside the (Drop) Box

Dhiru Kholia, Openwall and University of British Columbia; Przemysław Węgrzyn, CodePainters

Dropbox is the leading cloud-based file storage system. It is Python based and challenging to reverse engineer. Dhiru Kholia was able to use open source and novel methods to unpackage, decrypt, and gain access to the source code of Dropbox. In doing so, he realized that two main pieces of information exist: host_id and host_int. Host_id is a 128-bit persistent secret value, created during registration, and is not affected by password changes. It is also stored in cleartext (although newer versions of Dropbox have stored host_id in an encrypted fashion). Host_int can be sniffed or requested from the Dropbox server once the host_id is known.

Dropbox also offers a quick launch browser version that requires two-factor authentication upon first login. This two-factor authentication can be bypassed once Dhiru figured out the internal APIs. He showed a demo of unpacking and hijacking. Dhiru has developed a module for Metasploit which works for older versions of Dropbox which logged sensitive information. The authors are working on a new Metasploit module.

Illuminating the Security Issues Surrounding Lights Out Server Management

Anthony J. Bonkoski, Russ Bielawski, and J. Alex Halderman, University of Michigan

Intelligence Platform Management Interface (IPMI) is an industry specification created by Intel that is integrated directly into each server and is intended to enable the automation needed to manage a large cluster. Typically, IPMI devices are embedded and most are Linux-based with a remote virtual console and high network connectivity over HTTP or SSL to the cluster. These IPMI devices are widespread and many are directly connected to the Internet. Anthony Bonkoski demonstrated that IPMI is a perfect spying backdoor; it is always on and often pre-enabled, contains powerful remote tools, and it's an embedded system where security was an afterthought. It comes preloaded with default passwords and anonymous accounts enabled, and it even stored the passwords in plaintext.

The authors looked at the SuperMicro product with firmware by ATEN. This product is riddled with security vulnerabilities, including very basic ones. All input validation and permission checking is done on the client side, so the server accepts everything as authentic and valid. The authors were able to perform shell injection, gaining root access. Buffer overflows were pervasive, including no bounds checking in even the basic access control. It also had little to no defenses against buffer overflows. It allowed for stack and heap execution, contained no stack canaries, and had limited address space layout randomization. In approximately seven minutes, the authors were able to carry out a buffer overflow attack and gain root access. The authors found approximately 40,000 SuperMicro devices that are suspected to be vulnerable.

Security fixes include never attaching IPMI to the Internet, changing default passwords and certifications, or disabling IPMI altogether. Vendors must do a better job of including security in these cluster management devices.

Low-Level Attacks

Summarized by Rik Farrow (rik@usenix.org)

"Weird Machines" in ELF: A Spotlight on the Underappreciated Metadata

Rebecca Shapiro, Sergey Bratus, and Sean W. Smith, Dartmouth College

Rebecca (bx) Shapiro began by saying that most of the defenses against exploits have focused on code and not on executables' metadata. There are defenses against data-driven attacks, like return-oriented programming (ROP), with ASLR being the best, and code-signing and input checking being less effective. The "weird machine " that she learned to exploit is the runtime loader (RTLD_START()) in libc.so. The runtime loader handles relocation of code, and uses the information in the ELF (Executable and Linkable Format) header to determine how to do this.

The authors created a tool named Cobbler, along with a simple language, BF, which builds relocation information that will result in code execution during the relocation phase of loading. Bx explained how a mov instruction works, then how a jump-non-zero branch works by looping over a list of relocation entries. Using their tool with a Ubuntu 11.10 toolchain, they created an exploit for the ping program, which is SUID and owned by root. The exploit replaces two system calls using just nine relocation entries and one symbol table entry, so it will execute any program that follows the "--type" option as root.

Someone asked how their attack compared to previous work, and bx answered that the previous redirection attack is much more targeted, but they do reference that work in their paper. The session chair said that what he really liked is that this work presents a new paradigm, where metadata must be checked as well as code.

Introducing Die Datenkrake: Programmable Logic for Hardware Security Analysis

Dmitry Nedospasov, FG SecT, TU Berlin; Thorsten Schröder, modzero AG

Thorsten Schröder told us that so far, there has been no plug-in arch for HW/embedded device security analysis. For example, if you want to sniff high-speed buses in real time, you need to decode and filter the data before sending it to the PC over a slower bus. Existing solutions include oscilloscopes, logic analyzers, microcontrollers, and FPGA modules you design yourself.

The DDK architecture combines a microcontroller with an FPGA, directly connected together with a 16-bit data bus between them. DDK is open source, with user-friendly interfaces and connectors, test pads, and breakout of GPIO pins, and the ARM microcontroller can be updated via USB. They use eight RJ45 connectors for I/O channels, with six GPIO pins to the FPGA via buffers to protect it, five volt and ground. You can control power to the FPGA, and there are breakout pins between the buffers and the FPGA. Their OS is based on the FreeRTOS, with the basic modules and framework written in Verilog. Internally, the Wishbone Bus is used to connect the core modules. Uses include capturing high-speed protocols, such as are used in PCs.

With DDK, you can do man-in-the-bus attacks. They can also be used to decode software-defined radio, and can monitor eight channels at once, filtering out just the data of interest. Schröder used this to capture and process a wireless keyboard using an A7125 radio module attached to the DDK. Another area is glitching, introducing non-invasive faults within single target clock cycles. They have plans to add high-speed storage and to improve performance.

Someone asked how this work compares to Ozman's board, and Schröder answered that he hadn't looked at it. Someone else asked if this was all open source. Schröder replied that everything is realized, including the Verilog code and schematics.

The Page-Fault Weird Machine: Lessons in Instruction-less Computation

Julian Bangert, Sergey Bratus, Rebecca Shapiro, and Sean W. Smith, Dartmouth College

Sergey Bratus began by saying that Julian Bangert had done most of the work, and that it had been nominated for a p3wn award. Classical techniques for hacking use "weird machines," usually part of the code under attack that is used instead to execute code of the attacker's choice. Sergey said that exploits are proofs-by-construction that these weird machines exist in code. But Bangert's research looks into using CPU state that is not directly exposed as a weird machine. The x86 memory management unit (MMU) is not just a big lookup table because it can also perform complex logic. The MMU also writes error codes to the stack, or where in memory the MMU believes the stack pointer is. By using traps or, in particular, page faults, it is possible to modify memory by using a carefully crafted set

of stored entries in the task state segment (TSS). Unlike the exploit described in the first paper in this session, the compiler (https://github.com/jbangert/trapcc) cannot exploit a Linux system, but it can run on a bare machine and run the Game of Life by changing memory.

Someone asked, what are the practical consequences of this, beyond just being able to use the MMU as a weird machine? Sergey responded that a binary that used this for obfuscation would be very interesting to debug. If a processor is using this computation for control flow, it would be difficult to debug. The emulator faults are the only area of interest currently. Why did we do this? You can't build a reference monitor for hardware unless you understand how the hardware actually works. But this not an exploit per se, just delimiting the space in which you can look for bugs. We advance by finding new types of weird machines. The more we look for computation in unlikely places, the more we learn. Trey Darley then pointed out that there is a video demo that shows the "Game of Life" on YouTube.