Black Hat USA 2016

Archived: 2026-04-06 01:55:24 UTC

briefings - august 3 & 4

Keynote

The Hidden Architecture of our Time: Why This Internet Worked How We Could Lose It and the Role Hackers Play

What we call the Internet, was not our first attempt at making a global data network that spanned the globe. It was just the first one that worked.

Why?

In this talk, I'll lay out what I see as how the Internet actually works. It's increasingly likely that there will be attempts to *change* the principles of the net, and the reality is that widespread hacking is the exact sort of force that brought us this working-ish system in the first place.

We need to talk about the values of cryptography, of open software and networks, of hackers being a force for measurable good. We need to talk about how infrastructure like DNS -- it was there 25 years ago, we can imagine it will be there 25 years from now -- acts as foundation for future development in a way that the API of the hour doesn't.

Things do need to be better, and we need to talk about the role of Government in that. The things that need to be better are technical in nature, and guide research priorities that are outright not being addressed at present.

Essentially, I'd like to provide a model for comprehending the Internet as it stands, that prevents harm to it (how much could we have used EC2 if SSH was illegal) while providing the useful resources to promote its continued operation.

We can't keep screwing this up forever. NTIA has noted half (!) of the population warily backing away. Let's talk about how it really works, so we can discuss how we can do it better.

Briefings

$hell on Earth: From Browser to System Compromise

The winning submissions to Pwn2Own 2016 provided unprecedented insight into the state of the art in software exploitation. Every successful submission provided remote code execution as the super user (SYSTEM/root) via the browser or a default browser plugin. In most cases, these privileges were attained by exploiting the Microsoft Windows or Apple OS X kernel. Kernel exploitation using the browser as an initial vector was a rare sight in previous contests.

This presentation will detail the eight winning browser to super user exploitation chains (21 total vulnerabilities) demonstrated at this year's Pwn2Own contest. We will cover topics such as modern browser exploitation, the complexity of kernel Use-After-Free exploitation, and the simplicity of exploiting logic errors and directory traversals in the kernel. We will analyze all attack vectors, root causes, exploitation techniques, and possible remediations for the vulnerabilities presented.

Reducing attack surfaces with application sandboxing is a step in the right direction, but the attack surface remains expansive and sandboxes are clearly still just a speed bump on the road to complete compromise. Kernel exploitation is clearly a problem which has not disappeared and is possibly on the rise. If you're like us, you can't get enough of it; it's shell on earth.

1000 Ways to Die in Mobile OAuth

OAuth has become a highly influential protocol due to its swift and wide adoption in the industry. The initial objective of the protocol was specific: it serves the authorization needs for websites. However, the protocol has been significantly repurposed and re-targeted over the years: (1) all major identity providers, e.g., Facebook, Google and Microsoft, have re-purposed OAuth for user authentication; (2) developers have re-targeted OAuth to the mobile platforms, in addition to the traditional web platform. Therefore, we believe that it is necessary and timely to conduct an in-depth study to demystify OAuth for mobile application developers.

Our work consists of two pillars: (1) an in-house study of the OAuth protocol documentation that aims to identify what might be ambiguous or unspecified for mobile developers; (2) a field-study of over 600 popular mobile applications that highlights how well developers fulfill the authentication and authorization goals in practice. The result is really worrisome: among the 149 applications that use OAuth, 89 of them (59.7%) were incorrectly implemented and thus vulnerable. In the paper, we pinpoint the key portions in each OAuth protocol flow that are security critical, but are confusing or unspecified for mobile application developers. We then show several representative cases to concretely explain how real implementations fell into these pitfalls. Our findings have been communicated to vendors of the vulnerable applications. Most vendors positively confirmed the issues, and some have applied fixes. We summarize lessons learned from the study, hoping to provoke further thoughts about clear guidelines for OAuth usage in mobile applications

A Journey from JNDI/LDAP Manipulation to Remote Code Execution Dream Land

JNDI (Java Naming and Directory Interface) is a Java API that allows clients to discover and look up data and objects via a name. These objects can be stored in different naming or directory services such as RMI, CORBA, LDAP, or DNS.

This talk will present a new type of vulnerability named "JNDI Reference Injection" found on malware samples attacking Java Applets (CVE-2015-4902). The same principles can be applied to attack web applications running JNDI lookups on names controlled by attackers. As we will demo during the talk, attackers will be able to use different techniques to run arbitrary code on the server performing JNDI lookups.

The talk will first present the basics of this new vulnerability including the underlying technology, and will then explain in depth the different ways an attacker can exploit it using different vectors and services. We will focus on exploiting RMI, LDAP and CORBA services as these are present in almost every Enterprise application.

LDAP offers an alternative attack vector where attackers not able to influence the address of an LDAP lookup operation may still be able to modify the LDAP directory in order to store objects that will execute arbitrary code upon retrieval by the application lookup operation. This may be exploited through LDAP manipulation or simply by modifying LDAP entries as some Enterprise directories allow.

A Lightbulb Worm?

Could a worm spread through a smart light network? This talk explores the idea, and in particular dives into the internals of the Philips Hue smart light system, and details what security has been deployed to prevent this.

Examples of hacking various aspects of the system are presented, including how to bypass encrypted bootloaders to read sensitive information. Details on the firmware in multiple versions of the Philips Hue smart lamps and bridges are discussed. This talk concentrates on examples of advanced techniques used in attacking IoT/embedded hardware devices.

A Retrospective on the Use of Export Cryptography

TLS has experienced three major vulnerabilities stemming from "export-grade" cryptography in the last year---FREAK, Logajm, and Drown. Although regulations limiting the strength of cryptography that could be exported from the United States were lifted in 1999, and export ciphers were subsequently deprecated in TLS 1.1, Internet-

wide scanning showed that support for various forms of export cryptography remained widespread, and that attacks exploiting export-grade cryptography to attack non-export connections affected up to 37% of browser-trusted HTTPS servers in 2015. In this talk, I'll examine the technical details and historical background for all three export-related vulnerabilities, and provide recent vulnerability measurement data gathered from over a year Internet-wide scans, finding that 2% of browser-trusted IPv4 servers remain vulnerable to FREAK, 1% to Logjam, and 16% to Drown. I'll examine why these vulnerabilities happened, how the inclusion of weakened cryptography in a protocol impacts security, and how to better design and implement cryptographic protocols in the future. Having been involved in the discovery of all three export vulnerabilities, I'll distill some lessons learned from measuring and analyzing export cryptography into recommendations for technologists and policymakers alike, and provide a historical context for the current "going dark" and Apple vs. FBI debate.

Abusing Bleeding Edge Web Standards for AppSec Glory

Through cooperation between browser vendors and standards bodies in the recent past, numerous standards have been created to enforce stronger client-side control for web applications. As web appsec practitioners continue to shift from mitigating vulnerabilities to implementing proactive controls, each new standard adds another layer of defense for attack patterns previously accepted as risks. With the most basic controls complete, attention is shifting toward mitigating more complex threats. As a result of the drive to control for these threats client-side, standards such as SubResource Integrity (SRI), Content Security Policy (CSP), and HTTP Public Key Pinning (HPKP) carry larger implementation risks than others such as HTTP Strict Transport Security (HSTS). Builders supporting legacy applications actively make trade-offs between implementing the latest standards versus accepting risks simply because of the increased risks newer web standards pose.

In this talk, we'll strictly explore the risks posed by SRI, CSP, and HPKP; demonstrate effective mitigation strategies and compromises which may make these standards more accessible to builders and defenders supporting legacy applications; as well as examine emergent properties of standards such as HPKP to cover previously unforeseen scenarios. As a bonus for the breakers, we'll explore and demonstrate exploitations of the emergent risks in these more volatile standards, to include multiple vulnerabilities uncovered quite literally during our research for this talk (which will hopefully be mitigated by d-day).

Access Keys Will Kill You Before You Kill the Password

AWS users, whether they are devops in a startup or system administrators tasked with migrating an enterprise service into the cloud, interact on a daily basis with the AWS APIs, using either the web console or tools such as the AWS CLI to manage their infrastructure. When working with the latter, authentication is done using long-lived

access keys that are often stored in plaintext files, shared between developers, and sometimes publicly exposed. This creates a significant security risk as possession of such credentials provides unconditional and permanent access to the AWS API, which may yield catastrophic events in case of credentials compromise. This talk will detail how MFA may be consistently required for all users, regardless of the authentication method. Furthermore, this talk will introduce several open-source tools, including the release of one new tool, that may be used to allow painless work when MFA-protected API access is enforced in an AWS account.

Account Jumping Post Infection Persistency & Lateral Movement in AWS

The widespread adoption of AWS as an enterprise platform for storage, computing and services makes it a lucrative opportunity for the development of AWS focused APTs. We will cover pre-infection, post-infection and advanced persistency techniques on AWS that allows an attacker to access staging and production environments, as well as read and write data and even reverse its way from the cloud to the the corporate datacenter.

This session will cover several methods of infection including a new concept - "account jumping" for taking over both PaaS (e.g. ElasticBeans) and IaaS (EC2, EC2 Containers) resources, discussing poisoned AMIs, dirty account transfer, as well as leveraging S3 and CloudFront for performing AWS specific credentials thefts that can easily lead to full account access. We will then discuss the post-infection phase and how attackers can manipulate AWS resources (public endpoints like EC2 IPS, Elastic IPS, load balancers and more) for complete MITM attacks on services. We will demonstrate how attackers code can be well hidden via Lambda functions, some cross zone replication configuration and the problem with storage affinity to a specific account. We'll examine hybrid deployments from the cloud and compromising the on premise datacenter by leveraging and modifying connectivity methods (HW/SW VPN, Direct connect or cloud hub). Finally, we'll end with a discussion on best practices that can be used to protect from such attacks such as bastion SSH/RDP gateways, understanding the value of CASB based solutions and where they fit, leverage audit and HSM capabilities in AWS as well as looking at different Isolation approaches to create isolation between administrators and the cloud while still providing access to critical services.

Adaptive Kernel Live Patching: An Open Collaborative Effort to Ameliorate Android N-Day Root Exploits

Although 0-day exploits are dangerous, we have to admit that the largest threat for Android users are kernel vulnerabilities that have been disclosed but remain unfixed. Having been in the spotlight for weeks or even months, these kernel vulnerabilities usually have clear and stable exploits; therefore, underground businesses commonly utilize them in malware and APTs. The reason for the long periods of remaining unfixed is complex,

partly due to the time-consuming patching and verification procedures, or possibly because the vendors care more about innovating new products than securing existing devices. As such, there are still a lot devices all over the world subject to root attacks. The different patching status of various vendors causes fragmentation, and vendors usually don't provide the exact up-to-date kernel source code for all devices, so it is extremely difficult to patch vulnerable devices in scale. We will provide stats of the current Android kernel vulnerability landscape, including the device model population and the corresponding vulnerability rates. Some vulnerabilities with great impact but slow fixing progress will be discussed. The whole community strives to solve this problem, but obviously this cannot be done discretely with limited hands.

In this talk, we present an adaptive Android kernel live patching framework, which enables open and live patching for kernels. It has the following advantages: (1) It enables online hotpatching without interrupting user-experience. Unlike existing Linux kernel hotpatching solutions, it works directly on binaries and can automatically adjust to different device models with different Android kernel versions. (2) It enables third party vendors, who may not access the exact source code of the device kernel and drivers, to perform live patching. (3) Except for the binary patching scheme, it also provides a Lua based patching scheme, which makes patch generation and delivery even easier. It also has stronger confinement. This framework saves developers from repeating the tedious and error-prone patch porting work, and patches can be provided from various vendors, thus the patch deployment period can be greatly shortened. Only offering the power to perform adaptive live patching is not enough -- we need to regulate it just in case the hotpatches introduce further vulnerabilities and backdoors. So, a special alliance with membership qualification is formed. Only those selected vendors can provide patches and audit patches submitted from other alliance members. Furthermore, we will build a reputation ranking system for the patch providers, a mechanism similar to app stores. The Lua based patching scheme can provide even more restrictive regulations upon the operations of patches. Finally, this framework can be easily extended and applied to general Linux platforms. We believe that improving the security of the whole ecosystem is not a dream of our own. We call for more and more parties to join in this effort to fight the evils together.

Advanced CAN Injection Techniques for Vehicle Networks

The end goal of a remote attack against a vehicle is physical control, usually by injecting CAN messages onto the vehicle's network. However, there are often many limitations on what actions the vehicle can be forced to perform when injecting CAN messages. While an attacker may be able to easily change the speedometer while the car is driving, she may not be able to disable the brakes or turn the steering wheel unless the car she is driving meets certain prerequisites, such as traveling below a certain speed. In this talk, we discuss how physical, safety critical systems react to injected CAN messages and how these systems are often resilient to this type of manipulation. We will outline new methods of CAN message injection which can bypass many of these restrictions and demonstrate the results on the braking, steering, and acceleration systems of an automobile. We end by suggesting ways these systems could be made even more robust in future vehicles.

AirBnBeware: Short Term Rentals Long Term Pwnage

What's scarier, letting HD Moore rent your house and use your home network for day or being the very next renter that uses that network? With the colossal growth of the vacation rental market over the last five years (AirBnb, HomeAway), travellers are now more vulnerable than ever to network based attacks targeted at stealing personal information or outright pwnage. In 2006, the security industry desperately warned of the dangers of using public Wi-Fi at coffee shops. In 2010, we reshaped the conversation around the frightful security of Internet provided at hotels. And now, in 2016, we will start a new battle cry against the abysmal state of network security enabled by short term rentals. Both renters and property owners have a serious stake in this game. Whether you're renting a room in a foreign city to attend a conference or you're profiting off of your own empty domicile, serious risks abound: MitM traffic hi-jacking, accessing illegal content, device exploitation, and more. Common attacks and their corresponding defenses (conventional or otherwise) will be discussed, with a strong emphasis on practicality and simplicity. This talk will contain demos of attacks, introduce atypical hardware for defense, and encourage audience participation.

AMSI: How Windows 10 Plans to Stop Script-Based Attacks and How Well It Does It

In Windows 10, Microsoft introduced the AntiMalware Scan Interface (AMSI) which is designed to target script-based attacks and malware. Script-based attacks have been lethal for enterprise security and with advent of PowerShell, such attacks have become increasingly common. AMSI targets malicious scripts written in PowerShell, VBScript, JScript etc. and drastically improves detection and blocking rate of malicious scripts. When a piece of code is submitted for execution to the scripting host, AMSI steps in and the code is scanned for malicious content. What makes AMSI effective is, no matter how obfuscated the code is, it needs to be presented to the script host in clear text and unobfuscated. Moreover, since the code is submitted to AMSI just before execution, it doesn't matter if the code came from disk, memory or was entered interactively. AMSI is an open interface and MS says any application will be able to call its APIs. Currently, Windows Defender uses it on Windows 10. Has Microsoft finally killed script-based attacks? What are the ways out? The talk will be full of live demonstrations.

An AI Approach to Malware Similarity Analysis: Mapping the Malware Genome With a Deep Neural Network

In recent years, cyber defenders protecting enterprise networks have started incorporating malware code sharing identification tools into their workflows. These tools compare new malware samples to a large databases of known malware samples, in order to identify samples with shared code relationships. When unknown malware binaries are found to share code "fingerprints" with malware from known adversaries, they provides a key clue into which adversary is generating these new binaries, thus helping develop a general mitigation strategy against that family of threats. The efficacy of code sharing identification systems is demonstrated every day, as new family of threats are discovered, and countermeasures are rapidly developed for them.

Unfortunately, these systems are hard to maintain, deploy, and adapt to evolving threats. First and foremost, these systems do not learn to adapt to new malware obfuscation strategies, meaning they will continuously fall out of date with adversary tradecraft, requiring, periodically, a manually intensive tuning in order to adjust the formulae used for similarity between malware. In addition, these systems require an up to date, well maintained database of recent threats in order to provide relevant results. Such a database is difficult to deploy, and hard and expensive to maintain for smaller organizations. In order to address these issues we developed a new malware similarity detection approach. This approach, not only significantly reduces the need for manual tuning of the similarity formulate, but also allows for significantly smaller deployment footprint and provides significant increase in accuracy. Our family/similarity detection system is the first to use deep neural networks for code sharing identification, automatically learning to see through adversary tradecraft, thereby staying up to date with adversary evolution. Using traditional string similarity features our approach increased accuracy by 10%, from 65% to 75%. Using an advanced set of features that we specifically designed for malware classification, our approach has 98% accuracy. In this presentation we describe how our method works, why it is able to significantly improve upon current approaches, and how this approach can be easily adapted and tuned to individual/organization needs of the attendees.

An Inconvenient Trust: User Attitudes Toward Security and Usability Tradeoffs for Key-Directory Encryption Systems

Many critical communications now take place digitally, but recent revelations demonstrate that these communications can often be intercepted. To achieve true message privacy, users need end-to-end message encryption, in which the communications service provider is not able to decrypt the content. Historically, end-to-end encryption has proven extremely difficult for people to use correctly, but recently tools like Apple's iMessage and Google's End-to-End have made it more broadly accessible by using key-directory services. These tools (and others like them) sacrifice some security properties for convenience, which alarms some security experts, but little is known about how average users evaluate these tradeoffs. In a 52-person interview study, we asked participants to complete encryption tasks using both a traditional key-exchange model and a key-directory-based registration model. We also described the security properties of each (varying the order of presentation) and asked participants for their opinions. We found that participants understood the two models well and made coherent assessments about when different tradeoffs might be appropriate. Our participants recognized that the less-convenient

exchange model was more secure overall, but found the security of the registration model to be "good enough" for many everyday purposes.

An Insider's Guide to Cyber-Insurance and Security Guarantees

$75 billion. That's the amount of money businesses, governments, and individuals pay every year to security companies. While some security companies provide good value, the reality is the number of incidents are still getting worse and more frequent. Hundreds of millions of people have had their personal information stolen, businesses all over the world are losing intellectual property, and financial fraud is in the billions of dollars. These stories are constant, seemingly never-ending, and customers are tired of it. They are even apathetic to the degree that customers are turning to cyber-insurance as an alternative to breach prevention. We know this because cyber-insurance is a thing. In fact, cyber-insurance is a skyrocketing business that is already influencing every area of the information security industry. This rise of cyber-insurance has also provided a new way for security vendors to help their customers. A way for them to make a real positive impact, differentiate themselves, and align their incentives to that of their own customers - I'm talking about security guarantees.

Security guarantees or guaranteeing security is almost a taboo subject in the industry. As skeptics are quick to point out, nothing is 100% secure. Everything can be hacked. They're technically right, of course, but they're also missing the bigger picture. Just like we all buy electronics, cars, tools, or toys for the kids, all of these items sometimes break - yet, every manufacturer still provides some kind of guarantee. Most often, at least a replacement, a manufacture can do this because they know how often their product breaks. If every other major industry in the world can do it, the security industry can too! And while many InfoSec practitioners are not yet aware of this, a few security vendors are already offering security guarantees. From private conversations, at least a half dozen or more are actively working with cyber-insurers and creating security guarantee programs of their own. Many of our peers are investing their time in this space as well. In not too long, security guarantees will become common.

InfoSec practitioners who want to get a head start, or even a leg up, in cyber-insurance and security guarantees - this presentation is just for you. Also, one does not simply launch a security guarantee program. A great many things must be discussed, analyzed, and accounted for first. The business model of the program must be carefully designed, product efficacy must be measured, risk calculated, lawyers consulted, impact on financial accounting rules understood, liability reinsured, and more. Security vendors, if you're interested in how to go about creating a security guarantee program of your own, I'll be providing several helpful tools and a process. And business managers who would like to understand the landscape and how security guarantees are a great help in the purchase process, this talk is also for you.

Analysis of the Attack Surface of Windows 10 Virtualization-Based Security

In Windows 10, Microsoft introduced virtualization-based security (VBS), the set of security solutions based on a hypervisor. In this presentation, we will talk about details of VBS implementation and assess the attack surface - it is very different from other virtualization solutions. We will focus on the potential issues resulting from the underlying platform complexity (UEFI firmware being a primary example).

Besides a lot of theory, we will also demonstrate actual exploits: one against VBS itself and one against vulnerable firmware. The former is non-critical (provides bypass of one of VBS features), the latter is critical.

Before attending, one is encouraged to review the two related talks from Black Hat USA 2015: "Battle of the SKM and IUM: How Windows 10 Rewrites OS Architecture" and "Defeating Pass-the-Hash: Separation of Powers."

Applied Machine Learning for Data Exfil and Other Fun Topics

Machine learning techniques have been gaining significant traction in a variety of industries in recent years, and the security industry is no exception to it's influence. These techniques, when applied correctly, can help assist in many data driven tasks to provide interesting insights and decision recommendations to analyst. While these techniques can be powerful, for the researchers and analyst who are not well versed in machine learning, there can exist a gap in understanding that may prevent them from looking at and applying these tools to problems machine learning techniques could assist with.

The goal of this presentation is to help researchers, analyst, and security enthusiast get their hands dirty applying machine learning to security problems. We will walk the entire pipeline from idea to functioning tool on several diverse security related problems, including offensive and defensive use cases for machine learning. Through these examples and demonstrations, we will be able to explain in a very concrete fashion every step involved to tie in machine learning to the specified problem. In addition, we will be releasing every tool built, along with source code and related datasets, to enable those in attendance to reproduce the research and examples on their own. Machine learning based tools that will be released with this talk include an advanced obfuscation tool for data exfiltration, a network mapper, and command and control panel identification module.

Attacking SDN Infrastructure: Are We Ready for the Next-Gen Networking?

Software-Defined Networking (SDN), by decoupling the control logic from the closed and proprietary implementations of traditional network devices, allows researchers and practitioners to design new innovative

network functions/protocols in a much easier, more flexible, and powerful way. This technology has gained significant attentions from both industry and academia, and it is now at its adoption stage. When considering the adoption of SDN, the security vulnerability assessment is an important process that must be conducted against any system before the deployment and arguably the starting point toward making it more secure.

In this briefing, we explore the attack surface of SDN by actually attacking each layer of SDN stack. The SDN stack is generally composed of control plane, control channel and data plane: The control plane implementations, which are commonly known as SDN controllers or Network OS, implementations are commonly developed and distributed as an open-source project. Of those various Network OS implementations, we attack the most prevalent ones, OpenDaylight (ODL) [1] and Open Network Operating System (ONOS) [2]. These Network OS projects are both actively led by major telecommunication and networking companies, and some of the companies have already deployed them to their private cloud or network [3, 4]. For the control channel, we also attack a well-known SDN protocol [5], OpenFlow. In the case of the data plane, we test some OpenFlow-enabled switch device products from major vendors, such as HP and Pica8.

Of the attacks that we disclose in this briefing, we demonstrate some of the most critical attacks that directly affect the network (service) availability or confidentiality. For example, one of the attack arbitrarily uninstalls crucial SDN applications running on an ODL(or ONOS) cluster, such as routing, forwarding, or even security service applications. Another attack directly manipulates logical network topology maintained by an ODL(or ONOS) cluster to cause network failures. In addition, we also introduce some of the SDN security projects. We briefly go over the design and implementation of Project Delta, which is an official open-source SDN penetration testing tool pushed forward by Open Networking Foundation Security group, and Security-Mode ONOS, a security extension that protects the core of ONOS from the possible threats of untrusted third-party applications.

References [1] Medved, Jan, et al. "Opendaylight: Towards a model-driven sdn controller architecture." 2014 IEEE 15th International Symposium on. IEEE, 2014. [2] Berde, Pankaj, et al. "ONOS: towards an open, distributed SDN OS."Proceedings of the third workshop on Hot topics in software defined networking. ACM, 2014. [3] Jain, Sushant, et al. "B4: Experience with a globally-deployed software defined WAN." ACM SIGCOMM Computer Communication Review. Vol. 43. No. 4. ACM, 2013. [4] CORD: Reinventing Central Offices for Efficiency and Agility. http://opencord.org (2016). [5] OpenFlow. OpenFlow Switch Specification version 1.1.0. Tech. rep., 2011. http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf.

Augmenting Static Analysis Using Pintool: Ablation

Ablation is a tool built to extract information from a process as it executes. This information is then imported into the disassembly environment where it used to resolve virtual calls, highlight regions of code executed, or visually diff samples. The goal of Ablation is to augment static analysis with minimal overhead or user interaction.

C++ binaries can be a real pain to audit sometimes due to virtual calls. Instead of having to reverse class, object, and inheritance relationships, Ablation can resolve any observed virtual calls, and create fully interactive x-refs in

IDA; Disassembled C++ reads like C!

When augmenting analysis by importing runtime data, much of the information is displayed using a color scheme. This allows the info to be passively absorbed making it useful, rather than obtrusive.

Ablation makes it simple to diff samples by and highlight where the samples diverge. This is achieved by comparing the code executed rather than just comparing data. Consider comparing a heavily mutated crash sample, and the source sample. The root cause of the crash is normally tedious and unrewarding. Using Ablation, the root cause can often be determined simply by running each sample, and using the appropriate color scheme. This also means that visualizing the code coverage of a sample set becomes as simple as running each.

Recent findings have indicated that highly traversed code is not particularly interesting, and code infrequently executed or adjacent is more interesting. Ablation could be used to identify undocumented features in a product given a sample set.

Vulnerability research is all about the details. Having this information passively displayed could be the difference between confusion and discovery. Ablation will be made open source at BH2016.

AVLeak: Fingerprinting Antivirus Emulators for Advanced Malware Evasion

AVLeak is a tool for fingerprinting consumer antivirus emulators through automated black box testing. AVLeak can be used to extract fingerprints from AV emulators that may be used by malware to detect that it is being analyzed and subsequently evade detection, including environmental artifacts, OS API behavioral inconsistencies, emulation of network connectivity, timing inconsistencies, process introspection, and CPU emulator "red pills."

Emulator fingerprints may be discovered through painstaking binary reverse engineering, or with time consuming black box testing using binaries that conditionally choose to behave benignly or drop malware based on the emulated environment. AVLeak significantly advances upon prior approaches to black box testing, allowing researchers to extract emulator fingerprints in just a few seconds, and to script out testing using powerful APIs.

AVLeak will be demoed live, showing real world fingerprints discovered using the tool that can be used to detect and evade popular consumer AVs including Kaspersky, Bitdefender engine (licensed out to 20+ other AV products), AVG, and VBA. This survey of emulation detection methods is the most comprehensive examination of the topic ever presented in one place.

Bad for Enterprise: Attacking BYOD Enterprise Mobile Security Solutions

The global market for Bring Your Own Device (BYOD) and enterprise mobility is expected to quadruple in size over the next four years, hitting $284 billion by 2019. BYOD software is used by some of the largest organizations and governments around the world. Barclays, Walmart, AT&T, Vodafone, United States Department of Homeland

Security, United States Army, Australian Department of Environment and numerous other organizations, big and small, all over the world. Enterprise Mobile Security (EMS) is a component of BYOD solutions that promises data, device and communications security for enterprises. Amongst others, it aims to solve Data Loss, Network Privacy and jailbreaking/rooting of devices.

Using the Good Technology EMS suite as an example, my talk will show that EMS solutions are largely ineffective and in some cases can even expose an organization to unexpected risks. I will show attacks against EMS protected apps on jailbroken and non-jailbroken devices, putting to rest the rebuttal that CxOs and solution vendors often give penetration testers, "We do not support jailbroken devices." I will also introduce a groundbreaking tool, Swizzler, to help penetration testers confronted with apps wrapped into EMS protections. The tool conveniently automates a large amount of attacks that allows pen-testers to bypass each of the protections that Good and similar solutions implement. In a live demonstration of Swizzler I will show how to disable tampering detection mechanisms and application locks, intercept & decrypt encrypted data, and route "secure" HTTP requests through BURP into established Good VPN tunnels to attack servers on an organization's internal network. Swizzler will be released to the world along with my talk at Blackhat USA. Whether you are a CxO, administrator or user, you can't afford not to understand the risks associated with BYOD.

BadTunnel: How Do I Get Big Brother Power?

This presentation will introduce a new threat model. Based on this threat model, we found a flaw in the Windows system. It affects all Windows released in the last two decades, including Windows 10. It also has a very wide range of attacks surface. The attack can be performed on all versions of Internet Explorer, Edge, Microsoft Office, many third-party software, USB flash drives, and even Web server. When this flaw is triggered, YOU ARE BEING WATCHED.

We will also show you how to defend against this threat, particularly on those systems are no longer supported by Microsoft.

---

badWPAD

WPAD (Web Proxy Auto Discovery) is a protocol that allows computers to automatically discover Web proxy configurations. It is primarily used in networks where clients are only allowed to communicate to the outside through a proxy. The WPAD protocol has been around for almost 20 years (RFC draft 1999-07-28), but has well-known risks to it that have been largely ignored by the security community. This session will present the results of several experiments highlighting the flaws inherent to this badly designed protocol (WPAD), and bring attention to the many ways in which they can be easily exploited. Our research expands on these known flaws and proves a

surprisingly broad applicability of "badWPAD" for possible malicious use today by testing it in different environments. The speaker will share how his team initially deployed a WPAD experiment to test whether WPAD was still problematic or had been fixed by most software and OS vendors. This experiment included attacks in 1) Intranets and open-access networks (e.g. Free-WIFI spots and corporate networks) and 2) DNS attacks on clients leaking HTTP requests to the internet.

Attendees will hear the rather surprising results that this experiment yielded: The DNS portion of the experiment revealed more than 38 million requests to the WPAD honeypot domain names from oblivious customers - while the intranet Free-WIFI experiment proved that almost every second Wifi spot can be utilized as attack surface. This test included Wifi at airport lounges, conferences, hotel and on board of aircrafts, and were amazed that apparently nobody realized what their laptop was secretly requesting. It seems that this neglected WPAD flaw is growing, while it's commonly assumed to be fixed. The paper will be backed up by statistics and reveal why badWPAD remains to be a major security concern and what should be done to protect against this serious risk.

Behind the Scenes of iOS Security

With over a billion active devices and in-depth security protections spanning every layer from silicon to software, Apple works to advance the state of the art in mobile security with every release of iOS. We will discuss three iOS security mechanisms in unprecedented technical detail, offering the first public discussion of one of them new to iOS 10.

HomeKit, Auto Unlock and iCloud Keychain are three Apple technologies that handle exceptionally sensitive user data – controlling devices (including locks) in the user's home, the ability to unlock a user's Mac from an Apple Watch, and the user's passwords and credit card information, respectively. We will discuss the cryptographic design and implementation of our novel secure synchronization fabric which moves confidential data between devices without exposing it to Apple, while affording the user the ability to recover data in case of device loss.

Data Protection is the cryptographic system protecting user data on all iOS devices. We will discuss the Secure Enclave Processor present in iPhone 5S and later devices and explain how it enabled a new approach to Data Protection key derivation and brute force rate limiting within a small TCB, making no intermediate or derived keys available to the normal Application Processor.

Traditional browser-based vulnerabilities are becoming harder to exploit due to increasingly sophisticated mitigation techniques. We will discuss a unique JIT hardening mechanism in iOS 10 that makes the iOS Safari JIT a more difficult target.

Beyond the MCSE: Active Directory for the Security Professional

Active Directory (AD) is leveraged by 95% of the Fortune 1000 companies for its directory, authentication, and management capabilities. This means that both Red and Blue teams need to have a better understanding of Active Directory, it's security, how it's attacked, and how best to align defenses. This presentation covers key Active Directory components which are critical for security professionals to know in order to defend AD. Properly securing the enterprise means identifying and leveraging appropriate defensive technologies. The provided information is immediately useful and actionable in order to help organizations better secure their enterprise resources against attackers. Highlighted are areas attackers go after including some recently patched vulnerabilities and the exploited weaknesses. This includes the critical Kerberos vulnerability (MS14-068), Group Policy Man-in-the-Middle (MS15-011 & MS15-014) and how they take advantages of AD communication.

Some of the content covered:

- Differing views of Active Directory: admin, attacker, and infosec.
- The differences between forests and domains, including how multi-domain AD forests affect the security of the forest.
- Dig into trust relationships and the available security features describing how attack techniques are impacted by implementing these trust security features.
- AD database format, files, and object storage (including password data).
- Read-Only Domain Controllers (RODCs), security impact, and potential issues with RODC implementation.
- Key Domain Controller information and how attackers take advantage.
- Windows authentication protocols over the years and their weaknesses, including Microsoft's next-generation credential system, Microsoft Passport, and what it means for credential protection.
- Security posture differences between AD on-premises and in the cloud (Microsoft Azure AD vs Office 365).
- Key Active Directory security features in the latest Windows OS versions - the benefits and implementation challenges.

Let's go beyond the standard MCSE material and dive into how Active Directory works focusing on the key components and how they relate to enterprise security.

Blunting the Phisher's Spear: A Risk-Based Approach for Defining User Training and Awarding Administrative Privileges

Solving the "people problem" of cyber security requires us to understand why people fall victim to spear phishing. Unfortunately, the only proactive solution being used against spear phishing is user training and education. But,

judging from the number of continued breaches, training appears to be limited in its effectiveness. Today's leading cybersecurity training programs focus on hooking people in repeated simulated spear phishing attacks and then showing them the nuances in the emails they missed. This "gotcha game" presumes that users merely lack knowledge, and if they are told often enough and repeatedly shown what they lack, they would become better at spear phishing detection. This is akin to trying to teach people to drive by constantly causing accidents and then pointing out why they had an accident each time.

We propose a radical change to this "one-size-fits all" approach. Recent human factors researchthe Suspicion, Cognition, Automaticity Model (SCAM) [1]identifies a small set of factors that lead to individual phishing victimization. Using the SCAM, we propose the development of an employee Cyber Risk Index (CRI). Similar to how financial credit scores work, the CRI will provide security analysts the ability to pinpoint the weak-links in organizations and identify who is likely to fall victim, who needs training, how much training, and also what the training should focus on. The CRI will also allow security analysts to identify which users get administrative access, replacing the current mostly binary, role-based apportioning method, where individuals are given access based on their organizational role and responsibilities, with a system that is based on individuals' quantified cyber risk propensity. The CRI based approach we present will lead to individualized, cognitive-behavioral training and an evidence-based approach to awarding users' admin privileges. These are paradigm-changing solutions that will altogether improve individual cyber resilience and blunt the effectiveness of spear phishing.

Breaking FIDO: Are Exploits in There?

The state of authentication is in such disarray today that a black hat is no longer needed to wreak havoc. One avenue to authentication improvement is offered by the FIDO Alliance's open specifications built around public key cryptography. Does FIDO present a better mousetrap? Are there security soft spots for potential exploitation, such as man-in-the-middle attacks, exploits aimed at supporting architecture, or compromises targeting physical hardware? We will pinpoint where vulnerabilities are hidden in FIDO deployments, how difficult they are to exploit, and how enterprises and organizations can protect themselves.

Breaking Hardware-Enforced Security with Hypervisors

Hardware-Enforced Security is touted as the panacea solution to many modern computer security challenges. While certainly adding robust options to the defenders toolset, they are not without their own weaknesses. In this talk we will demonstrate how low-level technologies such as hypervisors can be used to subvert the claims of security made by these mechanisms. Specifically, we will show how a hypervisor rootkit can bypass Intel's Trusted Execution Environment (TXT) DRTM (dynamic root of trust measurement) and capture keys from Intel's

AES-NI instructions. These attacks against TXT and AES-NI have never been published before. Trusted computing has had a varied history, to include technologies such as Trusted Execution Technology (TXT), ARM TrustZone, and now Microsoft Isolated User Mode and Intel SGX. All of these technologies attempt to protect user data from privileged processes snooping or controlling execution. These technologies claim that no elevated process, whether kernel based, System Management Mode (SMM) based, or hypervisor based will be able to compromise the user's data and execution.

This presentation will highlight the age-old problem of misconfiguration of Intel TXT by exploiting a machine through the use of another Intel technology, the Type-1 hypervisor (VT-x). Problems with these technologies have surfaced not as design issues but during implementation. Whether there remains a hardware weakness where attestation keys can be compromised, or a software and hardware combination, such as exposed DMA that permits exfiltration, and sometimes modification, of user process memory. This presentation will highlight one of these implementation flaws as exhibited by the open source tBoot project and the underlying Intel TXT technology. Summation will offer defenses against all too often pitfalls when deploying these systems, including proper deployment design using sealed storage, remote attestation, and hardware hardening.

Breaking Kernel Address Space Layout Randomization (KASLR) with Intel TSX

Kernel hardening has been an important topic, as many applications and security mechanisms often consider the kernel their Trusted Computing Base (TCB). Among various hardening techniques, kernel address space layout randomization (KASLR) is the most effective and widely adopted technique that can practically mitigate various memory corruption vulnerabilities, such as buffer overflow and use-after-free. In principle, KASLR is secure as long as no memory disclosure vulnerability exists and high randomness is ensured. In this talk, we present a novel timing side-channel attack against KASLR, called DrK (De-randomizing Kernel address space), which can accurately, silently, and rapidly de-randomize the kernel memory layout by identifying page properties: unmapped, executable, or non-executable pages. DrK is based on a new hardware feature, Intel Transactional Synchronization Extension (TSX), which allows us to execute a transaction without interrupting the underlying operating system even when the transaction is aborted due to errors, such as access violation and page faults. In DrK, we turned this property into a timing channel that can accurately distinguish the mapping status (i.e., mapped versus unmapped) and execution status (i.e., executable versus non-executable) of the privileged address space. In addition to its surprising accuracy and precision, the DrK attack is not only universally applicable to all OSes, even under a virtualized environment, but also has no visible footprint, making it nearly impossible to be detected in practice. We demonstrate that DrK breaks the KASLR of all major OSes, including Windows, Linux, and OS X with near-perfect accuracy in a few seconds. Finally, we propose potential hardware modifications that can prevent or mitigate the DrK attack.

Breaking Payment Points of Interaction (POI)

The payment industry is becoming more driven by security standards. However, the corner stones are still broken even with the latest implementations of these payments systems, mainly due to focusing on the standards rather than security. The best example for that is the ability to bypass protections put in place by points of interaction (POI) devices, by simple modifying several files on the point of sale or manipulating the communication protocols. In this presentation, we will explain the main flaws and provide live demonstrations of several weaknesses on a widely used pinpad. We will not exploit the operating system of the pinpad, but actually bypass the application layer and the business logic protections, i.e. the crypto algorithm is secure, but everything around it is broken. As part of our demos, we will include EMV bypassing, avoiding PIN protections and scraping PANs from various channels.

Brute-Forcing Lockdown Harddrive PIN Codes

This presentation demonstrates a method of brute-forcing an AES-256 encrypted hard drive by spoofing the front-panel keyboard. In addition to tears into the internal design of the hard drive, and extends the work by J. Czarny & R. Rigo to validate the (in)security of any encrypted drive based on the MB86C311 chipset.

Building a Product Security Incident Response Team: Learnings from the Hivemind

You've received vulnerability reports in your application or product, now what? As a positive, there is an abundance of incident response guidance for network security and a number of companies that have published their Product Security Incident Response Team (PSIRT) process for customers at a high level. Yet there is a dearth of detailed resources on how to implement PSIRT processes for organizations that have realized that Stage 7 of the SDL process (Response). To not only build but maintain secure products, organizations need to create mechanisms enabling their incident response teams to receive and respond to product incident reports, effectively partnering with development teams, customer support, and communications teams.

This session will be targeted at small to medium companies that have small or overstretched security teams, and will share content and best practices to support these teams' product incident response programs. Attendees will be provided with templates and actionable recommendations based on successful best practices from multiple mature security response organizations.

Building Trust & Enabling Innovation for Voice Enabled IoT

Voice enabled technology provides developers with great innovation opportunities as well as risks. The Voice Privacy Alliance created a set of 39 Agile security stories specifically for voice enabled IoT products as part of the Voice Privacy Innovation Toolkit. These security stories help product owners and security developer focals bake security into their voice enabled products to save time, money and decrease incidents and reputation damage. This is a very practical, hands-on tool for developers that the Voice Privacy Alliance believes is needed to secure voice enabled technologies and promote innovation.

Call Me: Gathering Threat Intelligence on Telephony Scams to Detect Fraud

Robocalling, voice phishing and caller ID spoofing are common cybercrime techniques used to launch scam campaigns through the telephony channel that many people have long trusted. More than 660,000 online complaints regarding unwanted phone calls were recorded on the top six phone complaints websites in 2015. More reliable than online complaints, a telephony honeypot provides complete, accurate and timely information about unwanted phone calls across the United States. By tracking calling patterns in a large telephony honeypot receiving over 600,000 calls per month from more than 90,000 unique source phone numbers, we gathered threat intelligence in the telephony channel. Leveraging this data we developed a methodology to uniquely "fingerprint" bad actors hiding behind multiple phone numbers and detect them within the first few seconds of a call. Over several months, more than 100,000 calls were recorded and several millions call records analyzed to validate our methodology. Our results show that only a few bad actors are responsible for the majority of the spam and scam calls and that they can be quickly identified with high accuracy using features extracted from the audio. This discovery has major implications for law enforcement and businesses that are presently engaged in combatting the rise of telephony fraud.

Can You Trust Me Now? An Exploration into the Mobile Threat Landscape

Before we dive into specific mobile vulnerabilities and talk as if the end times are upon us, let us pop the stack and talk about how the mobile environment works as a whole. We will explore the assumptions and design paradigms of each player in the overall mobile space, along with the requirements and inheritance problems they face. The value of this approach is that it allows us to understand and couch the impacts and implications of all

mobile vulnerabilities, be it bugs existing today or theoretical future vulnerabilities. The approach also allows us to catalogue all the design assumptions made and search for any generalized logical flaws that could serve as a lynchpin to undermine the entirety of mobile security and trust.

This talk focuses on the entirety of the mobile ecosystem, from the hardware components to the operating systems to the networks they connect to. We will explore the core components across mobile vendors and operating systems, focusing on bugs, logic, and root problems that potentially effect all mobile devices. We will discuss the limitations of mobile trusted computing and what can be done to protect both your data and the devices your data reside on. From the specific perspectives of trusted computing and hardware integrity, there are a handful of smartphone hardware platforms on the market. OEMs are constrained to release devices based on selecting and trusting one of these platforms. If a skilled attacker can break trust at the hardware level, the entire device becomes compromised at a very basic (and largely undetectable) level. This talk is about how to break that trust.

CANSPY: A Platform for Auditing CAN Devices

In the past few years, several tools have been released allowing hobbyists to connect to CAN buses found in cars. This is welcomed as the CAN protocol is becoming the backbone for embedded computers found in smartcars. Its use is now even spreading outside the car through the OBD-II connector: usage-based policies from insurance companies, air-pollution control from law enforcement or engine diagnostics from smartphones for instance. Nonetheless, these tools will do no more than what professional tools from automobile manufacturers can do. In fact, they will do less as they do not have knowledge of upper-layer protocols.

Security auditors are used to dealing with this kind of situation: they reverse-engineer protocols before implementing them on top of their tool of choice. However, to be efficient at this, they need more than just being able to listen to or interact with what they are auditing. Precisely, they need to be able to intercept communications and block them, forward them or modify them on the fly. This is why, for example, a platform such as Burp Suite is popular when it comes to auditing web applications.

In this talk, we present CANSPY, a platform giving security auditors such capabilities when auditing CAN devices. Not only can it block, forward or modify CAN frames on the fly, it can do so autonomously with a set of rules or interactively using Ethernet and a packet manipulation framework such as Scapy. It is also worth noting that it was designed to be cheap and easy to build as it is mostly made of inexpensive COTS. Last but not least, we demonstrate its versatility by turning around a security issue usually considered when it comes to cars: instead of auditing an electronic control unit (ECU) through the OBD-II connector, we are going to partially emulate ECUs in order to audit a device that connects to this very connector.

Captain Hook: Pirating AVs to Bypass Exploit Mitigations

Put a low-level security researcher in front of hooking mechanisms and you get industry-wide vulnerability notifications, affecting security tools such as Anti-Virus, Anti-Exploitations and DLP, as well as non-security applications such as gaming and productivity tools. In this talk we reveal six(!) different security issues that we uncovered in various hooking engines. The vulnerabilities we found enable a threat actor to bypass the security measures of the underlying operating system. As we uncovered the vulnerabilities one-by-one we found them to impact commercial engines, such as Microsoft's Detours, open source engines such as EasyHook and proprietary engines such as those belonging to TrendMicro, Symantec, Kaspersky and about twenty others.

In this talk we'll survey the different vulnerabilities, and deep dive into a couple of those. In particular, we'll take a close look at a vulnerability appearing in the most popular commercial hooking engine of a large vendor. This vulnerability affects the most widespread productivity applications and forced the vendor to not only fix their engine, but also that their customers fix their applications prior to releasing the patch to the public. Finally, we'll demonstrate how security tools can be used as an intrusion channel for threat actors, ironically defeating security measures.

Capturing 0day Exploits with PERFectly Placed Hardware Traps

The security industry has gone to great lengths to make exploitation more difficult. Yet we continue to see weaponized exploits used in malware campaigns and targeted attacks capable of bypassing OS and vendor exploit mitigation strategies. Many of these newly deployed mitigations target code-reuse attacks like return-oriented-programming. Unfortunately, the reality is that once attackers have control over code execution it's only a matter of time before they can circumvent these defenses, as the recent rise of EMET bypasses illustrates. We propose a new strategy to raise the bar significantly. Our approach blocks exploits before they gain execution, preventing the opportunity to bypass mitigations.

This presentation introduces a new cross-platform, hardware-assisted Control-Flow Integrity (CFI) approach to mitigate control-flow hijack attacks on the Intel architecture. Prior research has demonstrated the effectiveness of leveraging processor-provided features such as the Performance Monitoring Unit (PMU) in order to trap various events for detecting ROP behaviors. We extend and generalize this approach by fine-tuning low-level processor features that enable us to insert a CFI policy to detect and prevent abnormal branches in real-time. Our promising results have shown this approach capable of protecting COTS binaries from control-flow hijack attempts stemming from use-after-free and memory corruption vulnerabilities with acceptable overhead on modern Windows and Linux systems.

In this talk, we will cover our research methodology, results, and limitations. We will highlight novel solutions to major obstacles we faced, including: proper tracking of Windows thread context swapping; configuration of PMU interrupt delivery without tripping Microsoft's PatchGuard; efficient algorithms for discovery of valid branch destinations in PE and ELF files at run-time; and the impact of operating in virtualized environments. The

effectiveness of our approach using hardware-assisted traps to monitor program execution and enforce CFI policies on mispredicted branches will be demonstrated in real-time. We will prevent weaponized exploits targeting Windows and Linux x86-64 operating systems that nominally bypass anti-exploit technologies like Microsoft's EMET tool. We will also present collected metrics on performance impact and the real-world applications of this technology.

Certificate Bypass: Hiding and Executing Malware from a Digitally Signed Executable

Malware developers are constantly looking for new ways to evade the detection and prevention capabilities of security solutions. In recent years, we have seen many different tools, such as packers and new encryption techniques, help malware reach this goal of hiding the malicious code. If the security solution cannot unpack the compressed or encrypted malicious content (or at least unpack it dynamically), then the security solution will not be able to identify that it is facing malware. To further complicate the matter, we present a new technique for hiding malware (encrypted and unencrypted) inside a digitally signed file (while still keeping the file with a valid certificate) and executing it from the memory, using a benign executable (which acts as a reflective EXE loader, written from scratch). Our research demonstrates our Certificate Bypass tool and the Reflective EXE Loader. During the presentation, we will focus on the research we conducted on the PE file structure. We will take a closer look at the certificate table and how we can inject data to the table without damaging the certificate itself (the file will still look and be treated as a valid digitally signed file). We will examine the tool we wrote to execute PE files from memory (without writing them to the disk). We will cover the relevant fields in the PE structure, as well as the steps required to run a PE file directly from the memory without requiring any files on disk. Last, we will conclude the demonstration with a live example and show how we bypass security solutions based on the way they look at the certificate table.

Crippling HTTPS with Unholy PAC

You're in a potentially malicious network (free WiFi, guest network, or maybe your own corporate LAN). You're a security conscious netizen so you restrict yourself to HTTPS (browsing to HSTS sites and/or using a "Force TLS/SSL" browser extension). All your traffic is protected from the first byte. Or is it?

We will demonstrate that, by forcing your browser/system to use a malicious PAC (Proxy AutoConfiguration) resource, it is possible to leak HTTPS URLs. We will explain how this affects the privacy of the user and how credentials/sessions can be stolen. We will present the concept of "PAC Malware" (a malware which is implemented only as Javascript logic in a PAC resource) that features: a 2-way communication channel between

the PAC malware and an external server, contextual phishing via messages, denial-of-service options, and sensitive data extraction from URI's. We present a comprehensive browser PAC feature matrix and elaborate more about this cross-platform (Linux, Windows, Mac) and cross-browser (IE, Chrome, Safari) threat.

Crumbling the Supercookie and Other Ways the FCC Protects Your Internet Traffic

You've probably heard of network neutrality. In 2015, the Federal Communications Commission enacted transformative rules that prohibit Internet service providers from blocking, throttling, or creating "fast lanes" for online content. The Open Internet Order protects your right to enjoy the lawful content, applications, services, and devices of your choosing. But it also empowers the FCC to protect the security and privacy of your Internet traffic. This talk will give an overview of the FCC's security and privacy authorities, which now cover broadband Internet service, as well as telephone, cable, and satellite connectivity. We will explain how the FCC investigates violations of federal communications law, and how it brings enforcement actions against offenders. In just the past two years, the FCC's Enforcement Bureau has initiated several high-profile law enforcement actions related to security and privacy. We required Verizon to stop injecting a unique identifier "supercookie" into third-party web requests, unless a customer consents. We also required AT&T and Cox to improve their customer information safeguards, after their security failures led to information on hundreds of thousands of customers getting unacceptably and unnecessarily exposed.*

Most recently, the FCC formally proposed new Internet security and privacy rules. The Commission recommended that, if your Internet service provider wants to share information from or about you, it should first obtain your affirmative, opt-in consent. We will explain how the rulemaking process functions, and how you can file comments on FCC proceedings. We will also leave time for a Q & A session. Whether you'd like to ask about net neutrality, robocalls, wifi router firmware (we know many of you have thoughts about that mixup!), or anything else communications related, this is your opportunity. In fact, you can even ask about your cable appointment we bet you didn't know the FCC has rules about that, too!

Cunning with CNG: Soliciting Secrets from Schannel

Secure Channel (Schannel) is Microsoft's standard SSL/TLS Library underpinning services like RDP, Outlook, Internet Explorer, Windows Update, SQL Server, LDAPS, Skype and many third party applications. Schannel has been the subject of scrutiny in the past several years from an external perspective due to reported vulnerabilities, including an RCE. What about the internals? How does Schannel guard its secrets?

This talk looks at how Schannel leverages Microsoft's CryptoAPI-NG (CNG) to cache the master keys, session keys, private and ephemeral keys, and session tickets used in TLS/SSL connections. It discusses the underlying data structures, and how to extract both the keys and other useful information that provides forensic context about connection. This information is then leveraged to decrypt a session that uses ephemeral key exchanges.

Information in the cache lives for at least 10 hours by default on modern configurations, storing up to 20,000 entries for client and server each. This makes it forensically relevant in cases where other evidence of the connection may have dissipated.

Cyber War in Perspective: Analysis from the Crisis in Ukraine

The conflict between Russia and Ukraine appears to have all the ingredients for "cyber war". Moscow and Kyiv are playing for the highest geopolitical stakes, and both countries have expertise in information technology and computer hacking. However, there are still many skeptics of cyber war, and more questions than answers. Malicious code is great for espionage and crime, but how much does it help soldiers on the battlefield? Does computer hacking have strategic effects? What are the political and military limits to digital operations in peacetime and war? This NATO-funded research project, undertaken by 20 leading authorities on national security and network security, is a benchmark for world leaders and system administrators alike, and sheds light on whether "cyber war" is now reality -- or still science fiction. Further, it helps decision makers to understand that national security choices today have ramifications for democracy and human rights tomorrow.

Dangerous Hare: Hanging Attribute References Hazards Due to Vendor Customization

For the purposes of tailoring the Android to different hardware platforms, countries/regions and other needs, hardware manufacturers (e.g. Qualcomm), device manufacturers, carriers and others have aggressively customized Android into thousands of system images. This practice has led to a highly fragmented ecosystem where the complicated relations among its components and apps though which one party interacts with the other have been seriously compromised. This leads to the pervasiveness of Hare (hanging attribute references e.g. package, activity, service action names, authorities and permissions), a type of vulnerabilities never investigated before.

In this talk, we will show that such flaws could have serious security implications, that is, a malicious app can acquire critical system capabilities by pretending to be the owner of an attribute who has been used on a device while the party defining it does not exist due to vendor customizations. On the factory image of 97 most popular Android devices, we discovered 21557 likely Hare flaws, demonstrating the significant impacts of the problem from stealing user's voice notes, controlling the screen unlock process, replacing Google Email's account settings to injecting messages into Facebook app and Skype. We will also show a set of new techniques we developed for automatically detecting Hare flaws within different Android versions, which can be utilized by the device manufacturers and other parties to secure their custom OSes. And we will provide the guidance for avoiding this pitfall when building future systems.

Dark Side of the DNS Force

DNS is an essential substrate of the Internet, responsible for translating user-friendly Internet names into machine-friendly IP addresses. Without DNS, it would be an impossible mission for us to navigate through the Internet. As we have seen in recent years, DNS-based attacks launched by adversaries remain a constant lethal threat in various forms. The record-breaking 300gbps DNS amplification DDoS attack against Spamhaus presented by Cloudflare at Black Hat 2013 is still vivid in our minds. Since then (in the last 3 years), thanks to the dark force's continuous innovations, the dark side of the DNS force is getting much more pernicious. Today, the dark side is capable of assembling an unprecedented massive attacking force of an unimaginable scale and magnitude. As an example, leveraging up to 10X of the Internet domain names, a modern DNS-based attack can easily take down any powerful online service, disrupt well-guarded critical infrastructure, and cripple the Internet, despite all the existing security postures and hardening techniques we have developed and deployed.

In this talk, we will present and discuss an array of new secret weapons behind the emerging DNS-based attacks from the dark side. We will analyze the root causes for the recent surges of the Internet domain counts from 300-million a year ago to over 2-billion. Some real use cases will be shown to illustrate the domain surges' impact on the Internet's availability and stability, especially with spikes up to 5-billion domains. We will focus on the evolution of random subdomain weapon which can generate a large number of queries to nonexistent fully qualified domain names such as 01mp5u89.arkhamnetwork.org and 01k5jj4u.arkhamnetwork.org to overload and knock down both authoritative name servers and cache servers along the query paths. Starting as a simple primitive tool used to disrupt competitors' gaming sites in order to win more users among the Chinese online gaming community about five years ago, random subdomain has become one of the most powerful disruptive weapons nowadays. As the attack targets move towards more high-profile and top level domains, the random subdomain weapon also becomes much sophisticated by blending attacking traffic with legitimate operations. It is a challenge for the cyber security community to distinguish bad traffic from benign ones in a cost-effective manner.

We will address this challenge by dissecting the core techniques and mechanisms used to boost attack strength and to evade detection. We will discuss techniques such as multiple level of random domains, mix use of constant names and random strings, innovative use of timestamps as unique domain names, as well as local and global escalations. We will demonstrate and compare different solutions for the accurate detection and effective mitigation of random subdomain and other active ongoing DNS-based attacks including DNS tunneling of data exfiltration on some most restricted networks due to the pervasiveness of DNS.

Defense at Hyperscale: Technologies and Policies for a Defensible Cyberspace

Cyber attackers have had the advantage for decades over defenders but we can and must change this with a more defensible cyberspace.

This talk describes the results of a recent task force to identify the top technologies, operational innovations and public policies which have delivered security at scale for the defense to catch up with attackers. All of these innovations have one thing in common: a dollar of defense buys far more than a dollar of offense. Now that we've recognized what has been most effective, the community has to repeat these successes at hyperscale, and the talk gives recommendations.

Demystifying the Secure Enclave Processor

The secure enclave processor (SEP) was introduced by Apple as part of the A7 SOC with the release of the iPhone 5S, most notably to support their fingerprint technology, Touch ID. SEP is designed as a security circuit configured to perform secure services for the rest of the SOC, with with no direct access from the main processor. In fact, the secure enclave processor runs it own fully functional operating system - dubbed SEPOS - with its own kernel, drivers, services, and applications. This isolated hardware design prevents an attacker from easily recovering sensitive data (such as fingerprint information and cryptographic keys) from an otherwise fully compromised device.

Despite almost three years have passed since its inception, little is still known about the inner workings of the SEP and its applications. The lack of public scrutiny in this space has consequently led to a number of misconceptions and false claims about the SEP.

In this presentation, we aim to shed some light on the secure enclave processor and SEPOS. In particular, we look at the hardware design and boot process of the secure enclave processor, as well as the SEPOS architecture itself. We also detail how the iOS kernel and the SEP exchange data using an elaborate mailbox mechanism, and how this data is handled by SEPOS and relayed to its services and applications. Last, but not least, we evaluate the SEP attack surface and highlight some of the findings of our research, including potential attack vectors.

Design Approaches for Security Automation

Organizations often scale at a faster pace than their security teams. Therefore, security teams need to deploy automation that can scale their processes. When it comes to your organization, what criteria should decide the best approach for security automation? Are there simpler alternatives to building a complex, custom built, automation environment? Where do you deploy? Which tools do you need? How do you ensure that your implementation will effectively enable teams versus just creating false positives at scale? This presentation will discuss criteria for

designing and evaluating security automation tools for your organization. The goal is provide audience members with effective small scale and large scale automation techniques for securing their environments.

Discovering and Exploiting Novel Security Vulnerabilities in Apple ZeroConf

With the proliferation of portable computing systems such as tablet, smartphone, Internet of Things (IoT), etc., ordinary users are facing the increasing burden to properly configure those devices, enabling them to work together. In response to this utility challenge, major device manufacturers and software vendors (e.g., Apple, Microsoft, Hewlett-Packard) tend to build their systems in a "plug-and-play" fashion, using techniques dubbed zero-configuration (ZeroConf). Such ZeroConf services are characterized by automatic IP selection, host name resolving and target service discovery. As the major proponent of ZeroConf techniques, Apple has adopted ZeroConf techniques in various frameworks and system services on iOS and OS X to minimize user involvements in system setup. However, when the design pendulum swings towards usability, concerns may arise whether the system has been adequately protected. In this presentation, we will report the first systematic study on the security implications of these ZeroConf techniques on Apple systems.

Our research brings to light a disturbing lack of security consideration in these systems' designs: major ZeroConf frameworks on the Apple platforms, including the Multipeer Connectivity and Bonjour, are mostly unprotected and system services, such as printer discovery and AirDrop, turn out to be completely vulnerable to an impersonation or Man-in-the-Middle (MitM) attack, even though attempts have been made to protect them against such threats. The consequences are serious, allowing a malicious device to steal documents to be printed out by other devices or files transferred between other devices. Most importantly, our study highlights the fundamental security challenges underlying ZeroConf techniques. Some of the vulnerabilities have not been fixed until this submission though we reported to Apple over half a year ago. We will introduce ZeroConf techniques and publish technical details of our attacks to Apple ZeroConf techniques. We will take Airdrop, Bonjour and Multipeer Connectivity as examples to show the vulnerabilities in their design and implementation and how we hacked these ZeroConf frameworks and system services to perform MitM attacks. We will also show that some of vulnerabilities are due to TLS' incompetence to secure device-to-device communication in the ZeroConf scenario, which is novel discovery and contributes to the state of the art.

Does Dropping USB Drives in Parking Lots and Other Places Really Work?

At every Black Hat you will inevitably hear hackers boasting that they can break into any company by dropping a malicious USB drive in the company's parking lot. This anecdote has even entered mainstream culture and was prominently featured in the Mr. Robot TV series. However despite its popularity, there has been no rigorous study of whether the attack works or is merely an urban legend. To answer this burning question and assess the actual threat posed by malicious USB drives, we dropped nearly 300 USB sticks on the University of Illinois Urbana-Champaign campus and measured who plugged in the drives. And oh boy how effective that was! Of the drives we dropped, 98% were picked up and for 48% of the drives, someone not only plugged in the drive but also

clicked on files. Join us for this talk if you are interested in physical security and want to learn more about the effectiveness of arguably the most well known anecdote of our community. We will provide an in-depth analysis of which factors influence users to pick up a drive, why users plug them in, and demo a new tool that can help mitigate USB attacks.

DPTrace: Dual Purpose Trace for Exploitability Analysis of Program Crashes

This research focuses on determining the practical exploitability of software issues by means of crash analysis. The target was not to automatically generate exploits, and not even to fully automate the entire process of crash analysis; but to provide a holistic feedback-oriented approach that augments a researcher's efforts in triaging the exploitability and impact of a program crash (or fault). The result is a semi-automated crash analysis framework that can speed-up the work of an exploit writer (analyst). Fuzzing, a powerful method for vulnerability discovery keeps getting more popular in all segments across the industry - from developers to bug hunters. With fuzzing frameworks becoming more sophisticated (and intelligent), the task of product security teams and exploit analysts to triage the constant influx of bug reports and associated crashes received from external researchers has increased dramatically. Exploit writers are also facing new challenges: with the advance of modern protection mechanisms, bug bounties and high-prices in vulnerabilities, their time to analyze a potential issue found and write a working exploits is shrinking.

Given the need to improve the existing tools and methodologies in the field of program crash analysis, our research speeds-up dealing with a vast corpus of crashes. We discuss existing problems, ideas and present our approach that is in essence a combination of backward and forward taint propagation systems. The idea here is to leverage both these approaches and to integrate them into one single framework that provides, at the moment of a crash, the mapping of the input areas that influence the crash situation and from the crash on, an analysis of the potential capabilities for achieving code execution. We discuss the concepts and the implementation of two functional tools developed by the authors (one of which was previously released) and go about the benefits of integrating them. Finally, we demonstrate the use of the integrated tool (DPTrace to be released as open-source at Black Hat) with public vulnerabilities (zero-days at the time of the released in the past), including a few that the authors themselves discovered, analyzed/exploited and reported.

Drone Attacks on Industrial Wireless: A New Front in Cyber Security

With new Drone technologies appearing in the consumer space daily, Industrial Plant operators are being forced to rethink their most fundamental assumptions about Industrial Wireless and Cyber-Physical security. This presentation will cover Electronic Threats, Electronic Defensive measures, Recent Electronic jamming incidents,

Latest Drone Threats and capabilities, defensive planning, and Electronic Attack Threats with Drones as delivery platform.

Dungeons Dragons and Security

The security community knows, the weak link is the human factor - from the project manager deciding that "security costs too much," to the operational bypassing its own company security measure, passing through the end user believing that nobody will ever think how he is using its cat's name as a password or a developper not following best practices.

We all arrive to the same conclusion - we need to train people to the computer security stakes. According to the author's experience, standard Security training is focused on the technical context (what a password is, how does a computer work etc.) and tends to bore or scare a neophyte audience.

This briefing will propose a new way to train a neophyte audience to the basic principles of Computer Security. The training is developed around a role playing game consisting in attacking and defending a building. A debriefing is done after the game to highlight all the similarities between the game and computer security stakes. The presentation will focus on the main feature of the training, and a white paper explaining how to conduct such a training will be available.

Exploiting Curiosity and Context: How to Make People Click on a Dangerous Link Despite Their Security Awareness

Messages containing links to malware-infected websites represent a serious threat. Despite the numerous user education efforts, people still click on suspicious links and attachments, and their motivations for clicking or not clicking remain hidden. We argue that knowing how people reason about their clicking behavior can help the defenders in devising more effective protection mechanisms. To this end, we report the results of two user studies where we sent to over 1600 university students an email or a Facebook message with a link from a non-existing person, claiming that the link leads to the pictures from the party last week. When clicked, the corresponding webpage showed the "access denied" message. We registered the click rates, and later sent to the participants a questionnaire that first assessed their security awareness, and then asked them about the reasons for their clicking behavior.

When addressed by first name, 56% of email and 38% of Facebook recipients clicked. When not addressed by first name, 20% of email and 42.5% of Facebook recipients clicked. Respondents of the survey reported high awareness of the fact that clicking on a link can have bad consequences (78%). However, statistical analysis

showed that this was not connected to their reported clicking behavior. By far the most frequent reason for clicking was curiosity about the content of the pictures (34%), followed by the explanations that the content or context of the message fits the current life situation of the person (27%), such as actually having been at a party with unknown people last week. Moreover, 16% thought that they know the sender. The most frequent reason for not clicking was unknown sender (51%), followed by the explanation that the message does not fit the context of the user (36%).

Therefore, it should be possible to make virtually any person click on a link, as any person will be curious about something, or interested in some topic, or find the message plausible because they know the sender, or because it fits their expectations (context). Expecting from the users error-free decision making under these circumstances seems to be highly unrealistic, even if they are provided with effective awareness training.

Moreover, while sending employees fake spear phishing messages from spoofed colleagues and bosses may increase their security awareness, it is also quite likely to have negative consequences in an organization. People's work effectiveness may decrease, as they will have to be suspicious of practically every message they receive. This may also seriously hamper social relationships within the organization, promoting the atmosphere of distrust. Thus, organizations need to carefully assess all pros and cons of increasing security awareness against spear phishing. In the long run, relying on technical in-depth defense may be a better solution, and more research and evidence is needed to determine the feasible level of defense that the non-expert users are able to achieve through security education and training.

GATTacking Bluetooth Smart Devices - Introducing a New BLE Proxy Tool

Bluetooth Low Energy is probably the most thriving technology implemented recently in all kinds of IoT devices: gadgets, wearables, smart homes, medical equipment and even banking tokens. The BLE specification assures secure connections through link-layer encryption, device whitelisting and bonding - a mechanisms not without flaws, although that's another story we are already aware of. A surprising number of devices do not (or simply cannot - because of the use scenario) utilize these mechanisms. The security (like authentication) is, in fact, provided on higher "application" (GATT protocol) layer of the data exchanged between the "master" (usually mobile phone) and peripheral device. The connection from "master" in such cases is initiated by scanning to a specific broadcast signal, which by design can be trivially spoofed. And guess what - the device GATT internals (so-called "services" and "characteristics") can also be easily cloned.

Using a few simple tricks, we can assure the victim will connect to our impersonator device instead of the original one, and then just proxy the traffic - without consent of the mobile app or device. And here it finally becomes interesting - just imagine how many attacks you might be able to perform with the possibility to actively intercept the BLE communication! Basing on several examples, I will demonstrate common flaws possible to exploit, including improper authentication, static passwords, not-so-random PRNG, excessive services, bad assumptions - which allow you to take over control of smart locks, disrupt smart home, and even get a free lunch. I will also suggest best practices to mitigate the attacks. Ladies and gentlemen - I give you the BLE MITM proxy. A free

open-source tool which opens a whole new chapter for your IoT device exploitation, reversing and debugging. Run it on a portable Raspberry Pi, carry around BLE-packed premises, share your experience and contribute to the code.

GreatFET: Making GoodFET Great Again

My evil plot began by making small but seemingly helpful contributions to the GoodFET project, a line of code here, a simple add-on board there. Soon I was answering the occasional question on IRC or the mailing list, and I was in: commit rights!

I had chosen my prey carefully. GoodFET, the preferred open source tool of discriminating hardware hackers around the world, consisted of too many disparate hardware designs. It was full of terrific ideas and PoCs, but it was becoming unmaintainable. The Facedancer variant alone had at least three different and incompatible code bases! The hardware designs were easy to build one at a time but needlessly costly for volume manufacturing. The project was ripe for a takeover.

I struck when Travis Goodspeed was most vulnerable, his faculties diminished by the hordes of Las Vegas. He accepted my $5. GoodFET was mine!

With GoodFET in my control I moved quickly to replace the entire project with something superior, something greater! Today I unleash GreatFET!

Hacking Next-Gen ATMs: From Capture to Cashout

Over the past year I have worked at understanding and breaking the new methods that ATM manufactures have implemented on producing "Next Generation" Secure ATM systems. This includes bypassing Anti-skimming/Anti-Shimming methods introduced to the latest generation ATMs, along with NFC long range attacks that allow real-time card communication over 400 miles away. This talk will demonstrate how a $2000 investment can perform unattended "cash outs," touching also on failures in the past with EMV implementations and how credit card data of the future will most likely be sold with the new EMV data - with a short life span. This talk will include a demonstration of "La-Cara," an automated cash out machine that works on current EMV and NFC ATMs. "La-Cara" is an entire fascia placed on the machine to hide the auto PIN keyboard and flashable EMV card system that silently withdraws money from harvested card data. This demonstration of the system can cash out around $20,000/$50,000 in 15 min. With these methods revealed we will be able to protect against similar types of attacks.

Hackproofing Oracle eBusiness Suite

A recent security review by David Litchfield of Oracle's eBusiness Suite (fully patched) revealed it is vulnerable to a number of (unauthenticated) remote code execution flaws, a slew of SQL injection vulnerabilities and Cross Site Scripting bugs. Used by large corporations across the globe the question becomes how does one secure this product given its weaknesses. This talk will examine those weakness with demonstration exploits then look at how one can protect their systems against these attacks.

Hardening AWS Environments and Automating Incident Response for AWS Compromises

Incident Response procedures differ in the cloud versus when performed in traditional, on-premise, environments. The cloud offers the ability to respond to an incident by programmatically collecting evidence and quarantining instances but with this programmatic ability comes the risk of a compromised API key. The risk of a compromised key can be mitigated but proper configuration and monitoring must be in place.

The talk discusses the paradigm of Incident Response in the cloud and introduces tools to automate the collection of forensic evidence of a compromised host. It highlights the need to properly configure an AWS environment and provides a tool to aid the configuration process.

Cloud IR How is it Different?

Incident response in the cloud is performed differently than when performed in on-premise systems. Specifically, in a cloud environment you can not walk up to the physical asset, clone the drive with a write-blocker, or perform any action that requires hands on time with the system in question. Incident response best practices advise following predefined practiced procedures when dealing with a security incident, but organizations moving infrastructure to the cloud may fail to realize the procedural differences in obtaining forensic evidence. Furthermore, while cloud providers produce documents on handling incident response in the cloud, these documents fail to address the newly released features or services that can aid incident response or help harden cloud infrastructure. (1.)

A survey of AWS facilities for automation around IR

The same features in cloud platforms that create the ability to globally deploy workloads in the blink of an eye can also add to ease of incident handling. An AWS user may establish API keys to use the AWS SDK to programmatically add or remove resources to an environment, scaling on demand. A savvy incident responder can

use the same AWS SDK, or (the AWS command line tools) to leverage cloud services to facilitate the collection of evidence. For example, using the AWS command line tools or the AWS SDK, a user can programmatically image the disk of a compromised machine with a single call. However, the power of the AWS SDK introduces a new threat in the event of an API key compromise.

Increased Attack Surface via Convenience (Walk through some compromise scenarios to illustrate)

There are many stories of users accidentally uploading their AWS keys to GitHub or another sharing service and then having to fight to regain control of the AWS account while their bill skyrockets. (2. 3.) And while these stories are sensational, they are preventable by placing limits on a cloud account directly. More concerning is the risk of a compromised key being used to access private data. A compromised API key without restrictions could access managed database, storage, or code repository services, to name a few. (4.) While the API key itself may not be used to access a targeted box, it is possible to use that key to clone a targeted box, and relaunch it with an attacker's SSH key, giving the attacker full access to the newly instantiated clone. While the consequences of a compromised API key can be dire, the risks can be substantially mitigated with proper configuration and monitoring.

Hardening of AWS Infrastructure
AWS environments can be hardened by following traditional security best practices and leveraging AWS services. AWS Services like CloudTrail and Config should be used to monitor and configure an AWS environment. CloudTrail provides logging of AWS API invocations tied to a specific API key. AWS Config provides historical insight into the configuration of AWS resources including users and the permissions granted in their policies.

API keys associated to AWS accounts should be delegated according to least privilege and therefore have the fewest number of permissions granted in its policy as possible. Furthermore, API keys should be tightened to restrict access only to the resources they need. Managing of these policies is made easier by the group and role constructs provided by AWS IAM, but it still leaves to the user having to understand each of the 195 policies currently recognized by IAM.

Introduction of Tools
We present custom tooling so the entire incident response process can be automated based on certain triggers within the AWS account. With very little configuration users could detect a security incident, acquire memory, take snapshots of disk images, quarantine, and have it presented to an examiner workstation all in the time it takes to get a cup of coffee.

Additional tooling is presented to aid in the recovery of an AWS account should a AWS key be compromised. The tool attempts to rotate compromised keys, identify and remove rogue EC2 instances and produce a report with next steps for the user.

Finally, we present a tool that examines an existing AWS environments and aides in configuring that environment to a hardened state. The tool recommends services to enable, permissions to remove from user accounts, and metrics to collect.

We discuss Incident Response in the cloud and introduce tools to automate the collection of forensic evidence of a compromised host. We highlight the need to properly configure an AWS environment and provide tools to aid the

configuration process.

References

1. AWS Security Resources. N.p., n.d. Web. 10 Apr. 2016.

.

2. Example AWS Key Compromises. Ed. Soulskill. N.p., n.d. Web. 10 Apr. 2016. .

3. IT News Article on AWS Keys. N.p., n.d. Web. 10 Apr. 2016. .

4. AWS Console Breach CloudSpaces. N.p., n.d. Web. 10 Apr. 2016. .

HEIST: HTTP Encrypted Information can be Stolen Through TCP-Windows

Over the last few years, a worryingly number of attacks against SSL/TLS and other secure channels have been discovered. Fortunately, at least from a defenders perspective, these attacks require an adversary capable of observing or manipulating network traffic. This prevented a wide and easy exploitation of these vulnerabilities. In contrast, we introduce HEIST, a set of techniques that allows us to carry out attacks against SSL/TLS purely in the browser. More generally, and surprisingly, with HEIST it becomes possible to exploit certain flaws in network protocols without having to sniff actual traffic. HEIST abuses weaknesses and subtleties in the browser, and the underlying HTTP, SSL/TLS, and TCP layers. Most importantly, we discover a side-channel attack that leaks the exact size of any cross-origin response. This side-channel abuses the way responses are sent at the TCP level. Combined with the fact that SSL/TLS lacks length-hiding capabilities, HEIST can directly infer the length of the plaintext message. Concretely, this means that compression-based attacks such as CRIME and BREACH can now be performed purely in the browser, by any malicious website or script, without requiring network access. Moreover, we also show that our length-exposing attacks can be used to obtain sensitive information from unwitting victims by abusing services on popular websites. Finally, we explore the reach and feasibility of exploiting HEIST. We show that attacks can be performed on virtually every web service, even when HTTP/2 is used. In fact, HTTP/2 allows for more damaging attack techniques, further increasing the impact of HEIST. In short, HEIST is a set of novel attack techniques that brings network-level attacks to the browser, posing an imminent threat to our online security and privacy.

Horse Pill: A New Type of Linux Rootkit

What if we took the underlying technical elements of Linux containers and used them for evil? The result a new kind rootkit, which is even able to infect and persist in systems with UEFI secure boot enabled, thanks to the way almost every Linux system boots. This works without a malicious kernel module and therefore works when kernel

module signing is used to prevent loading of unsigned kernel modules. The infected system has a nearly invisible backdoor that can be remote controlled via a covert network channel.

Hope is not lost, however! Come to the talk and see how the risk can be eliminated/mitigated. While this may poke a stick in the eye of the current state of boot security, we can fix it!

HTTP Cookie Hijacking in the Wild: Security and Privacy Implications

The widespread demand for online privacy, also fueled by widely-publicized demonstrations of session hijacking attacks against popular websites (see Firesheep), has spearheaded the increasing deployment of HTTPS. However, many websites still avoid ubiquitous encryption due to performance or compatibility issues. The prevailing approach in these cases is to force critical functionality and sensitive data access over encrypted connections, while allowing more innocuous functionality to be accessed over HTTP. In practice, this approach is prone to flaws that can expose sensitive information or functionality to third parties. In this work, we conduct an in-depth assessment of a diverse set of major websites and explore what functionality and information is exposed to attackers that have hijacked a user's HTTP cookies. We identify a recurring pattern across websites with partially deployed HTTPS; service personalization inadvertently results in the exposure of private information. The separation of functionality across multiple cookies with different scopes and inter-dependencies further complicates matters, as imprecise access control renders restricted account functionality accessible to non-session cookies. Our cookie hijacking study reveals a number of severe flaws; attackers can obtain the user's home and work address and visited websites from Google, Bing and Baidu expose the user's complete search history, and Yahoo allows attackers to extract the contact list and send emails from the user's account. Furthermore, e-commerce vendors such as Amazon and Ebay expose the user's purchase history (partial and full respectively), and almost every website exposes the user's name and email address. Ad networks like Doubleclick can also reveal pages the user has visited. To fully evaluate the practicality and extent of cookie hijacking, we explore multiple aspects of the online ecosystem, including mobile apps, browser security mechanisms, extensions and search bars. To estimate the extent of the threat, we run IRB-approved measurements on a subset of our university's public wireless network for 30 days, and detect over 282K accounts exposing the cookies required for our hijacking attacks. We also explore how users can protect themselves and find that, while mechanisms such as the EFF's HTTPS Everywhere extension can reduce the attack surface, HTTP cookies are still regularly exposed. The privacy implications of these attacks become even more alarming when considering how they can be used to deanonymize Tor users. Our measurements suggest that a significant portion of Tor users may currently be vulnerable to cookie hijacking.

HTTP/2 & QUIC - Teaching Good Protocols To Do Bad Things

The meteoric rise of SPDY, HTTP/2, and QUIC has gone largely unremarked upon by most of the security field. QUIC is an application-layer UDP-based protocol that multiplexes connections between endpoints at the application level, rather than the kernel level. HTTP/2 (H2) is a successor to SPDY, and multiplexes different HTTP streams within a single connection. More than 10% of the top 1 Million websites are already using some of these technologies, including much of the 10 highest traffic sites. Whether you multiplex out across connections with QUIC, or multiplex into fewer connections with HTTP/2, the world has changed. We have a strong sensation of Déjà vu with this work and our 2014 BlackHat USA MPTCP research. We find ourselves discussing a similar situation in new protocols with technology stacks evolving faster than ever before, and Network Security is largely unaware of the peril already upon it. This talk briefly introduces QUIC and HTTP/2, covers multiplexing attacks beyond MPTCP, discusses how you can use these techniques over QUIC and within HTTP/2, and discusses how to make sense of and defend against H2/QUIC traffic on your network. We will also demonstrate, and release, some tools with these techniques incorporated.

I Came to Drop Bombs: Auditing the Compression Algorithm Weapon Cache

A decompression bomb attack is relatively simple to perform --- but can be completely devastating to developers who have not taken the time to properly guard their applications against this type of denial of service. The decompression bomb is not a new attack - it's been around since at least 1996 - but unfortunately they are still horrifyingly common. The stereotypical bomb is the zip bomb, but in reality nearly any compression algorithm can provide fruit for this attack (images, HTTP streams, etc.). What algorithms have the highest compression ratio, the sloppiest parsers, and make for the best bomb candidates? This talk is about an ongoing project to answer that question. In addition to the compression algorithm audit, this research is generating a vast library of tools ("bombs") that can be used by security researchers and developers to test for this vulnerability in a wide variety of applications/protocols. These bombs are being released under an open-source license.

Into The Core - In-Depth Exploration of Windows 10 IoT Core

The Internet of Things is becoming a reality, and more and more devices are being introduced into the market every day. With this, the demand for technology that would ease device management, improve device security, and facilitate data analytics increases as well.

One such technology is Windows 10 IoT Core, Microsoft's operating system aimed at small footprint, low cost devices. It offers device servicing and manageability, enterprise grade security, and - combined with Microsoft's Azure platform - data analytics in the cloud. Given these features, Microsoft Windows 10 IoT Core will likely play a significant role in the future of IoT. As such, understanding how this operating system works on a deep

level is becoming important. Methods and techniques that would aid in assessing its security are also becoming essential.

In this talk I will first discuss the internals of the OS, including the security features and mitigations that it shares with the desktop edition. I will then enumerate the attack surface of a device running Windows 10 IoT Core as well as its potential susceptibility to malware. I will also talk about methods to assess the security of devices running Windows 10 IoT Core such as static/dynamic reverse engineering and fuzzing. I will end the talk with some recommendations on how to secure a Windows 10 IoT Core device.

Intra-Process Memory Protection for Applications on ARM and x86: Leveraging the ELF ABI

Today's software needs to isolate not only processes but the many components *within* a process from each other. Process-level isolation via jails, sandboxes, VMs, or hypervisors is finally becoming mainstream, but it misses an important point about modern software: its growing number of libraries that are all loaded into the same address space, and may all interact with complex inputs by way of vulnerable parsers. A process, even isolated, is as weak as the weakest of its components, but is as valuable as the most sensitive data it holds. Heartbleed was a perfect example of this: a faulty parser in a library could read absolutely everything in memory; there are many others less famous but no better. The biggest challenge of making intra-process memory protection practical is that it cannot require major changes to how software is written. A practical granular memory protection scheme must work for the existing C/C++ build chains, nor should it change the ABI. Further, it cannot rely on concepts that aren't already intuitively clear to C/C++ programmers. Many academic proposals for more granular memory access control stopped short of this. They disregard the glue what keeps the development process and runtime together: the ABI.

We demonstrate ELFbac, a system that uses the Linux ELF ABI to express access control policies between a program's components, such as libraries, and requires no changes to the GNU build chain. It enforces these policies by using a modified Linux loader and the Linux virtual memory system. ELFbac policies operate on the level of ELF object file sections. Custom data and code units can be created with existing GCC C/C++ attributes with a one-line annotation per unit; they are no more complex than C's static scoping. We have developed prototypes for ARM and x86. We used our ARM prototype to protect a validating proxy firewall for DNP3, a popular ICS protocol, and our x86 one to write a basic policy for Nginx. We will also demonstrate a policy for protecting OpenSSH.

Investigating DDOS - Architecture Actors and Attribution

DDOS attack usage has been accelerating, in terms of both attack volume and frequency. Such attacks present a major threat to enterprises worldwide. Presenters will discuss a number of novel techniques utilized by law enforcement and the private sector, to measure, study, and attribute attacks originating from sources such as embedded device botnets and booter/stresser services. Presenters will discuss the usage of honeypots to gather historical attack details, as well as best practices for conducting live DDOS attack testing. Representative PCAPs will be shown, dissected, and explain. Finally, presenters will provide examples of where these services are offered for sale, how they are purchased, and the individuals who operate them.

Iran's Soft-War for Internet Dominance

Over the past decade, the Islamic Republic of Iran has been targeted by continual intrusion campaigns from foreign actors that sought access to the country's nuclear facilities, economic infrastructure, military apparatus, and governmental institutions for the purpose of espionage and coercive diplomacy. Similarly, since the propagandic defacements of international communications platforms and political dissident sites conducted by an organization describing itself as the "Iranian Cyber Army" beginning in late 2009, Iranian actors have been attributed to a recurrent campaigns of intrusions and disruptions of private companies, foreign government entities, domestic opposition, regional adversaries and international critics. The intent of the CNO activities is not always discernable based on the tactics used or the data accessed, as the end implications of the disclosure of particular information is often distant and concealed. Where such intent is made evident, the reasons for Iranian intrusion campaigns range from retaliatory campaigns against adversaries, as a result of identifiable grievances, to surveillance of domestic opposition in support of the Islamic Republic establishment. Iranian intrusion campaigns have also reflected an interest in internal security operations against active political movements that have historically advocated for the secession of ethnic minority provinces or overthrow of the political establishment through violence. However, Iranian intrusion sets appear to be primarily interested in a broader field of challenges to the political and religious hegemony of the Islamic Republic. Previous reports on Iranian campaigns have referred to the targeting of Iranian dissident. However, in practice those targeted range from reformists operating within the establishment from inside of Iran to former political prisoners forced out of the country.

Across the records of hundreds of intrusion attempts of campaigns conducted by a distinct sets of actors, distinct patterns emerge in the types of individuals and organizations targeted by Iranian actors by internal security operations: high-profile individuals and organizations, such as journalists, human rights advocates or political figures, with extensive relationships and networks inside of Iran; members of the diplomatic establishment of Iran, and former governmental officials under previous administrations; adherents to non-Shia religions, participants in ethnic rights movements, or members of anti-Islamic Republic political organization; academics or public policy organizations critical of the Iranian government; cultural figures that promote values contrary to the interpretation of Islamic values promoted by the establishment; organizations fostering international collaboration and connections with the current Iranian administration; and international organizations conducting political programmes focused on Iran through funding by governmental agencies. In this presentation we will analyze in depth the results of several years of research and investigation on the intrusion activities of Iranian threat actors, particularly engaged in attacks against members of civil society.

Keystone Engine: Next Generation Assembler Framework

Assembler is an application that compiles a string of assembly code and returns instruction encodings. An assembler framework allows us to build new tools, and is a fundamental component in the Reverse Engineering (RE) toolset. However, a good assembler framework is sorely missed since the ice age! Indeed, there is no single multi-architecture, multi-platform and open source framework available and the whole RE community are badly suffering from this lingering issue.

We have decided to step up again to solve this challenge once and for all. We built Keystone, an assembler engine with unparalleled features:

- Multi-architecture, with support for Arm, Arm64 (AArch64/Armv8), Hexagon, Mips, PowerPC, Sparc, SystemZ, & X86 (include 16/32/64bit).
- Clean/simple/lightweight/intuitive architecture-neutral API.
- Implemented in C/C++ languages, with bindings for Python, NodeJS, Ruby, Go & Rust available.
- Native support for Windows & *nix (with Mac OSX, Linux, *BSD & Solaris confirmed).
- Thread-safe by design.
- Open source.

This talk is going to introduce some existing assembler frameworks, then goes into details of their design/implementation and explains their current issues. Next, we will present the architecture of Keystone and the challenges of designing and implementing it. The audience will understand the advantages of our engine and see why the future is assured, so that Keystone will keep getting better, stronger and become the ultimate assembler engine of choice for the security community.

Keystone aims to lay the ground for innovative works and open up new opportunities for future of security research and development. To conclude the talk, some new advanced RE tools built on top of Keystone will be introduced to demonstrate its power.

Keystone has a homepage at http://www.keystone-engine.org. Full source code of our engine will be released at Black Hat USA 2016.

Language Properties of Phone Scammers: Cyberdefense at the Level of the Human

The prevalence of human interactive components of serious system breaches continues to be a problem for every organization. Humans are the biggest vulnerability in any security system; helping people identify social engineering attempts over the phone will be cheaper and more effective than yet another technological implementation. At minimum it will add an important and necessary layer to defense in depth.
Forensic linguistics is the study of language as evidence for the law. It is a relatively new field and has not previously been applied to cybersecurity. Linguistic analysis uncovers several features of language interaction in a limited data set (recorded IRS phone scammers) that begin to answer how forensic linguistics could assist in cybersecurity defense.
This presentation will briefly introduce and explain polar tag questions, topic control, question deferral, and irregular narrative constructions in IRS scam phone calls, and offer some starting points for identifying such linguistic properties during the course of a phone call to help improve defense at the human level. We think this is only the beginning of applying forensic linguistics to cybersecurity.

Measuring Adversary Costs to Exploit Commercial Software: The Government-Bootstrapped Non-Profit C.I.T.L.

Many industries, provide consumers with data about the quality, content, and cost of ownership of products, but the software industry leaves consumers with very little data to act upon. In fact when it comes to how secure or weak a product is from a security perspective, there is no meaningful consumer facing data. There has long been a call for the establishment of an independent organization to address this need.

Last year, Mudge (from DARPA, Google, and L0pht fame) announced that after receiving a phone call from the White House he was leaving his senior position inside Google to create a non-profit organization to address this issue. This effort, known as CITL, is akin to Consumer Reports in its methodologies. While the media has dubbed it a "CyberUL", there is no focus on certifications or seals of approval, and no opaque evaluation metrics. Rather, like Consumer Reports, the goal is to evaluate software according to metrics and measurements that allow quantitative comparison and evaluation by anyone from a layperson, CFO, to security expert.

How? A wide range of heuristics that attackers use to identify which targets are hard or soft against new exploitation has been codified, refined, and enhanced. Some of these techniques are quite straightforward and even broadly known, while others are esoteric tradecraft. To date, no one has applied all of these metrics uniformly across an entire software ecosystem before and shared the results. For the first time, a peak at the Cyber Independent Testing Lab's metrics, methodologies, and preliminary results from assessing the software quality and inherent vulnerability in over 100,000 binary applications on Windows, Linux, and OS X will be revealed. All accomplished with binaries only.

Sometimes the more secure product is actually the cheaper, and quite often the security product is the most vulnerable. There are plenty of surprises like these that are finally revealed through quantified measurements. With this information, organizations and consumers can finally make informed purchasing decisions when it

comes the security of their products, and measurably realize more hardened environments. Insurance groups are already engaging CITL, as are organizations focused on consumer safety. Vendors will see how much better or worse their products are in comparison to their competitors. Even exploit developers have demonstrated that these results enable bug-bounty arbitrage.

That recommendation you made to your family members last holiday about which web browser they should use to stay safe (or that large purchase you made for your industrial control systems)? Well, you can finally see if you chose a hard or soft target… with the data to back it up.

Memory Forensics Using Virtual Machine Introspection for Cloud Computing

The relocation of systems and services into cloud environments is on the rise. Because of this trend users lose direct control over their machines and depend on the offered services from cloud providers. These services are especially in the field of digital forensics very rudimentary. The possibilities for users to analyze their virtual machines with forensic methods are very limited. In the underlying research of this talk a practical approach has been developed that gives the user additional capabilities in the field of forensic investigations. The solution focuses on a memory forensic service offering. To reach this goal, a management solution for cloud environments has been extended with memory forensic services. Self-developed memory forensic services, which are installed on each cloud node and are managed through the cloud management component, are the basis for this solution. Forensic data is gained via virtual machine introspection techniques. Compared to other approaches it is possible to get trustworthy data without influencing the running system. Additionally, a general overview about the underlying technologies is provided and the pros and cons are discussed. The solution approach is discussed in a generic way and practically implemented in a prototype. In this prototype OpenNebula is used for managing the cloud infrastructure in combination with Xen as virtualization component, LibVMI as Virtual Machine Introspection library and Volatility as forensic tool.

Next-Generation of Exploit Kit Detection by Building Simulated Obfuscators

Recently, driving-by downloads attacks have almost reached epidemic levels, and exploit-kit is the propulsion to signify the process of malware delivery. One of the key techniques used by exploit-kit to avoid firewall detection is obfuscating malicious JavaScript program. There exists an engine in each exploit kit, aka obfuscator, which transforms the malicious code to obfuscated code. Few researchers have studied obfuscation techniques utilized by exploit kit. Their main focus is on extracting information from the obfuscated page, such as common substring, common pattern, structure of the script (AST) and statistics of sensitive function invocation, and generating signatures. All of these studies are based on the analysis of obfuscated page, but not the obfuscator. One reason is that purchasing an obfuscator utilized by real exploit-kit is extremely expensive in the underground market. However, exploit-kit research can benefit from obfuscators in various aspects.

Our work rebuilds obfuscator for 6 notorious exploit kit families (Angler, Nuclear, Rig, Magnitude, Neutrino, SweetOrange). We will discuss our design to implement an obfuscator used by the exploit kit family, and evaluate how similar our obfuscator is to a real one. We would also like to open-source our obfuscator to benefit the research, which aims to provide better protection of the cyber-world. We performed a serial of experiences based on our obfuscators. With the obfuscator in hand, we are also able to generate more samples than we have ever observed, even those that haven't been created by real exploit-kit. We also simulate the evolution of obfuscator in each exploit kit family by building a new version upon the previous version. We derived some patterns on how obfuscator evolved and tent to predict what the next obfuscator variation could be. We also noticed that current variation naming convention may not properly reflect variation of exploit kit. Currently, people name a new variation of unknown sample by checking whether it shares the similar structure with existing samples. However, our experience shows that even a minor configuration file change in obfuscator could significantly change the obfuscated page. Therefore, we propose to use the actual change of obfuscator as the evidence to name a new variation. We also conduct an evaluation on how many times the obfuscator could amplify its change to the obfuscated page.

Nonce-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS

We investigate nonce-reuse issues with the Galois/Counter Mode (GCM) algorithm as used in TLS. Nonce reuse in GCM allows an attacker to recover the authentication key and forge messages as described by Joux. With an Internet-wide scan we identified over 70,000 HTTPS servers that are at risk of nonce reuse. We also identified 184 HTTPS servers repeating nonces directly in a short connection. Affected servers include large corporations, financial institutions, and a credit card company. We implement a proof of concept attack allowing us to violate the authenticity of affected HTTPS connections and inject content.

O-checker: Detection of Malicious Documents Through Deviation from File Format Specifications

Documents containing executable files are often used in targeted email attacks in Japan. We examine various document formats (Rich Text Format, Compound File Binary and Portable Document Format) for files used in targeted attacks from 2009 to 2012 in Japan. Almost all the examined document files contain executable files that ignore the document file format specifications. Therefore, we focus on deviations from file format specifications and examine stealth techniques for hiding executable files. We classify eight anomalous structures and create a tool named o-checker to detect them. O-checker detects 96.1% of the malicious files used in targeted email attacks in 2013 and 2014. There are far fewer stealth techniques than vulnerabilities of document processors. Additionally,

document file formats are more stable than document processors themselves. Accordingly, we assert that o-checker can continue detecting malware with a high detection rate for long periods.

OSS Security Maturity: Time to Put On Your Big Boy Pants!

Open source software (OSS) usage is on the rise and also continues to be a major source of risk for companies. OSS and 3rd party code may be inexpensive to use to build products but it comes with significant liability and maintenance costs. Even after high profile vulnerabilities in OpenSSL and other critical libraries, tracking and understanding exposure continues to challenge even at the most mature enterprise company. It doesn't matter if you are a software vendor or not, development and the use of OSS in your organization is most likely significant. It also doesn't matter if you have been developing software for years or are just getting started, or whether you have one product or one hundred, it can feel to many nearly impossible to keep up with OSS vulnerabilities or more important ensure they are properly mitigated.

This presentation looks at the real risk of using OSS and the best way to manage its use within your organization and more specifically the Product Development Lifecycle. We will examine all the current hype around OSS and separate out what are the real risks, and what organizations should be the most concerned about. We explore the true cost of using OSS and review the various factors that can be used to evaluate if a particular product or library should be used at your organization, including analyzing Vulnerability Metrics including Time to Patch. Getting your head wrapped around the issues and the need to improve OSS security is challenging, but then taking action at your organization can feel impossible. This presentation provides several real world examples that have been successful at a including: A case study of a single third party libraries vulnerability across several products will help to show why the result of investigating actual impact against your different products is valuable intelligence. We will provide learnings from your incident response function and why understanding the vulnerabilities in your current software can gain you valuable insight into creating smarter products to avoid maintenance costs. Finally, we will introduce a customized OSS Maturity Model and walk through the stages of maturity for organization developing software with regards to how they prioritize and internalize the risk presented by OSS.

Ouroboros: Tearing Xen Hypervisor with the Snake

The Xen Project has been a widely used virtualization platform powering some of the largest clouds in production today.
Sitting directly on the hardware below any operating systems, the Xen hypervisor is responsible for the management of CPU/MMU and guest operating systems.

Guest operating systems cound be controled to run in PV mode using paravirtualization technologies or HVM mode using hardware-assisted virtualization technologies.

Compare to HVM mode, PV mode guest OS kernel could recognize the existence of hypervisor and, thus, work normally via hypervisor inferfaces which are called hypercalls. While performing priviledged operations, PV mode guest OS would submit requests via hypercalls then the hypervisor do these operations for it after verifying its requests.

Inspired by Ouroboros, an ancient symbol with a snake bitting its tail, our team has found a critical verification bypass bug in Xen hypervisor and that will be used to tear the hypervisor a hole. With sepecific exploition vectors and payloads, malicious PV guest OS could control not only the hypervisor but also all other guest operating systems running on current platform.

Over the Edge: Silently Owning Windows 10's Secure Browser

Memory deduplication, a well-known technique to reduce the memory footprint across virtual machines, is now also a default-on feature inside the Windows 10 operating system. Deduplication maps multiple identical copies of a physical page onto a single shared copy with copy-on-write semantics. As a result, a write to such a shared page triggers a page fault and is thus measurably slower than a write to a normal page.

We leverage this side channel to build a weird machine and read arbitrary data in the system from the browser. By controlling the alignment and reuse of data in memory, we perform a byte-by-byte disclosure of high-entropy sensitive data, such as 64-bit code pointers randomized by ASLR. Next, even without control over data alignment or reuse, we show how to disclose randomized 64-bit heap pointers using a novel birthday attack. To show these attack primitives are practical, we have built an end-to-end JavaScript-based exploit against the new Microsoft Edge browser, in absence of software vulnerabilities and with all defenses turned on. Our exploit combines our deduplication-based primitives with a reliable Rowhammer attack to gain arbitrary memory read and write access in the browser.

Pangu 9 Internals

Pangu 9, the first (and only) untethered jailbreak tool for iOS 9, exploited a sequence of vulnerabilities in the iOS userland to achieve final arbitrary code execution in the kernel and persistent code signing bypass. Although these vulnerabilities were fixed in iOS 9.2, there are no details disclosed. This talk will reveal the internals of Pangu 9. Specifically, this talk will first present a logical error in a system service that is exploitable by any container app through XPC communication to gain arbitrary file read/write as mobile. Next, this talk will explain how Pangu 9 gains arbitrary code execution outside the sandbox through the system debugging feature. This talk will then elaborate a vulnerability in the process of loading the dyld_shared_cache file that enables Pangu 9 to achieve persistent code signing bypass. Finally, this talk will present a vulnerability in the backup-restore process that

allows apps signed by a revoked enterprise certificate to execute without the need of the user's explicit approval of the certificate.

Pay No Attention to That Hacker Behind the Curtain: A Look Inside the Black Hat Network

Each year thousands of security professionals answer the siren song of Black Hat USA. They come to learn from the best trainers, and the smartest(and best looking) speakers. And hey, this is Vegas, and when you're in Vegas, you make it rain...exploits.

Yes, every year thousands of security pros learn the latest tactics and techniques from the sharpest minds in the industry, and once they have, they can't wait to test them on the closest network they can find, our network. This presentation will help you understand what's going on behind the scenes at Black Hat. Who's attacking who? What are they doing? And what makes it all tick.

So come see what goes into standing up, and maintaining one of the most hostile networks on the planet. We'll share everything we can about the history of the network, the infrastructure we're using today, and the traffic patterns that keep us sweating, and laughing, well into the night.

PINdemonium: A DBI-Based Generic Unpacker for Windows Executable

Nowadays malware authors employ multiple obfuscation and packing techniques to hinder the process of reverse engineering and bypass the anti-virus (AV) signature based analysis. This is a significant threat for end user's PCs since this voids part of the AV analysis, and it is also a problem for professional reverse engineers that have to invest lot of time in order to unpack and study a single packed malware sample. The problem of unpacking is well studied in literature and several works have been proposed both for enhancing the end user's protection and supporting the malware analysts in their work. Different approaches exist in order to build a generic unpacker: debuggers, kernel modules, hypervisor modules, dynamic binary instrumentation (DBI). In this thesis we explore the possibility to exploit the functionality of a DBI framework since it provides great functionality useful during the analysis process: it allows an instruction level granularity inspection and modification, through high level APIs, which gives the analyst full control of the program being instrumented. Our system can extract and reconstruct the original program from a packed version of it, helping and speeding up the analysis of an obfuscated binary. The packers employ different techniques with various levels of complexity, but all of them must share one common behavior during the run-time unpacking: they have to write new code in memory and eventually execute it. Starting from this we have designed a generic unpacking algorithm that can correctly detect this behaviour and defeat the most popular of packing techniques. Not only the packing strategy can be really different, but the obfuscation can be increased by hiding the function imported by the program which is usually a valuable source of information during the process of reverse engineer. These are known in literature as Import Address Table (IAT) obfuscation techniques. Our tool tries to reconstruct a working PE from its packed version,

taking care of modern packing techniques like unpacking on dynamic memory allocated areas and tries to defeat the most used IAT obfuscation techniques.

In order to validate our work we have conducted two experiments. The first one demonstrate the generality of our unpacking process with respect to fifteen different packers. The second experiment demonstrates the effectiveness of our system against malware samples packed with both known and unknown packers. Our system was able to reconstruct a working unpacked binary for 63\% of the collected samples. When it is not possible to reconstruct a fully working PE, we provide all the memory dumps, representing the unpacked program along with a log about the unpacking process, which can be really useful to a malware analyst in order to speed up his work as it has been useful for us during the development of this tool. The source code of our tool can be found at https://github.com/Seba0691/PINdemonium.

PLC-Blaster: A Worm Living Solely in the PLC

We will present and demonstrate the first PLC only worm. Our PLC worm will scan and compromise Siemens Simatic S7-1200 v1-v3 PLCs without any external support. No PCs or additional hardware is required. The worm is fully self-contained and "lives" only on the PLC. The Siemens Simatic PLCs are managed using a proprietary Siemens protocol. Using this protocol the PLC may be stopped, started and diagnostic information may be read. Futhermore this protocol is used to upload and download user programs to the PLC. The older S7-300 and S7-400 PLCs are supported by several OpenSource solutions supporting the protocols used on these older PLCs. With the introduction of the S7-1200 the protocol has been replaced by a new version. We inspected the protocol based on the S7-1200v3 and implemented the protocol by ourselves. We are now able to install and extract any user program on these PLCs currently sold by Siemens. The current versions S7-1200v4 and S7-1500 again changed the protocol and are not susceptible to the attack.

Based on this work we developed a PLC program which scans a local network for other S7-1200v3 PLCs. Once these are found the program compromises these PLCs by uploading itself to these devices. The already installed user software is not removed and still running on the PLC. Our malware attaches itself to the original software and runs in parallel to the original user program. The operator does not notice any changed behavior. We developed the first PLC only worm. The worm is only written using the programming language SCL and does not need any additional support. For the remote administration of the compromised PLCs we implemented a Command&Control server. Infected PLCs automatically contact the C&C server and may be remotely controlled using this connection. Using this connection we can manipulate any physical input or output of the PLC. An additional proxy function enables us to access any additional system using a tunnel. Lastly the Stop mode may be initiated through the C&C connection requiring a cold restart of the PLC by disconnecting the power supply. We will demonstrate the attack during the talk.

Our worm prevents its detection and analysis. If the operator connects to the PLC using the programming software TIA Portal 11 the operator may notice unnamed additional function blocks. But when accessing these blocks the TIA Portal crashes preventing the forensic analysis. The infection of the PLC takes roughly 10 seconds. While the

infection is in progress the PLC is in Stop mode. As soon as the infection has succeeded the PLC undergoes a warm restart and the worm is running additionally to to the original user program. Our worm malware requires 38,5kb RAM and 216,6kb persistent memory. If the PLC does not offer the memory required by the original user software including our worm the worm may overwrite the original user program. Based on the actually used model of the S7-1200 different setups may be required.

Model RAM (Worm) Persistent Memory (Worm) S7-1211 50kb (77%) 1Mb (21%)
S7-1212 75kb (51%) 1MB (5 %)
S7-1214 100kb (38%) 4MB (5 %)
S7-1215 125kb (30%) 4MB (5 %)
S7-1217 150kb (25%) 4MB (5 %)

A critical requirement for the execution of a PLC program is the cycle time for one full cycle of the user program. Our malware requires 7ms per cycle. This is just 4.7% of the maximum cycle time configured by default on the PLC models we inspected. The original user program still has plenty of time to run. By default all Siemens Simatic S7-1200v1-v3 PLCs are susceptible to this attack. The PLC user programs may be uploaded and downloaded without any restriction. The Siemens Simatic PLCs support several protection mechanisms. We will explain these mechanisms and their result on the attack.

With the introduction of the S7-1200v4 Siemens introduced again a new protocol. These PLCs are not susceptible to the attack. The built-in copy protection restricts the user program to run only on a subset of PLCs with specific serial numbers. This protection is only implemented within the programming software (Siemens Simatic TIA Portal) used to install the software. We can upload and download user programs using this feature to any PLC using our own implementation. The whole protection is implemented on the client. This is the first time this is publicly shown. The built-in know-how protection forbids modifications of the user program on the PLC and prevents the extraction of the user program from the PLC. Again this protection is implemented only in the programming software (Siemens Simatic TIA Portal). Our own implementation can extract the user program, display the source code, modify the program and reinstall the modified program. This feature does not offer the protection advertised. This is the first time publicly shown. The built-in access protection does prevent the attack we will demonstrate. While we present an attack via the ethernet interface the installation of the user program can also happen using the field bus interface. Using this interface even PLCs not connected to the ethernet network may be compromised. Once the first PLC is infected using the Ethernet all other PLCs connected by the field bus would be compromised as well. This talk emphasizes the significance of the built in protection features in modern PLCs and their correct deployment by the user.

Pwning Your Java Messaging with Deserialization Vulnerabilities

Messaging can be found everywhere. It's used by your favourite Mobile Messenger as well as in your bank's backend system. Message Brokers such as Pivotal's RabbitMQ, IBM's WebSphere MQ and others often form a key component of a modern backend system's architecture. Furthermore, there are various messaging standards in

place like AMQP, MQTT, and STOMP. When it comes to the Java World it is rather unknown that Messaging in the Java ecosystem relies heavily on Java's serialization. Recent advances in the exploitation of Java deserialization vulnerabilities can be applied to exploit applications using Java messaging. This talk will show the attack surface of various Java messaging API implementations and their deserialization vulnerabilities. Last but not least, the Java Messaging Exploitation Tool (JMET) will be presented to help you identify and exploit message-consuming systems like a boss.

Recover a RSA Private Key from a TLS Session with Perfect Forward Secrecy

They always taught us that the only thing that can be pulled out from a SSL/TLS session using strong authentication and latest Perferct Forward Secrecy ciphersuites is the public key of the certificate exchanged during the handshake - an insufficient condition to place a MiTM attack without to generate alarms on the validity of the TLS connection and certificate itself. Anyway, this is not always true. In certain circumstances it is possible to derive the private key of server regardless of the size of the used modulus. Even RSA keys of 4096 bits can be factored at the cost of a few CPU cycles and computational resources. All that needed is the generation of a faulty digital signature from server, an event that can be observed when occurring certain conditions such as CPU overheating, RAM errors or other hardware faults. Because of these premises, devices like firewall, switch, router and other embedded appliances are more exposed than traditional IT servers or clients. During the talk, the author will explain the theory behind the attack, how common the factors are that make it possible and his custom pratical implementation of the technique. At the end, a proof-of-concept, able to work both in passive mode (i.e. only by sniffing the network traffic) and in active mode (namely, by participating directly in the establishment of TLS handshakes), will be released.

Samsung Pay: Tokenized Numbers Flaws and Issues

Samsung announced many layers of security to its Pay app. Without storing or sharing any type of user's credit card information, Samsung Pay is trying to become one of the most secure approaches offering functionality and simplicity for its customers. This app is a complex mechanism which has some limitations relating security. Using random tokenize numbers and implementing Magnetic Secure Transmission (MST) technology, which do not guarantee that every token generated with Samsung Pay would be applied to make a purchase with the same Samsung device. That means that an attacker could steal a token from a Samsung Pay device and use it without restrictions. Inconvenient but practical is that Samsung's users could utilize the app in airplane mode. This makes it impossible for Samsung Pay to have a full control process of the tokens pile. Even when the tokens have their own restrictions, the tokenization process gets weaker after the app generates the first token relating a specific card. How random is a Spay tokenized number? It is really necessary to understand how the tokens heretically

share similarities in the generation process, and how this affect the end users' security. What are the odds to guess the next tokenized number knowing the previous one?

Secure Penetration Testing Operations: Demonstrated Weaknesses in Learning Material and Tools

Following previous presentations on the dangers penetration testers face in using current off-the-shelf tools and practices, this presentation explores how widely available learning materials used to train penetration testers lead to inadequate protection of client data and penetration testing operations. With widely available books and other training resources targeting the smallest set of prerequisites, in order to attract the largest audience, many penetration testers adopt the techniques used in simplified examples to real world tests, where the network environment can be much more dangerous. Malicious threat actors are incentivized to attack and compromise penetration testers, and given current practices, can do so easily and with dramatic impact. This presentation will include a live demonstration of techniques for hijacking a penetration tester's normal practices, as well as guidance for examining and securing your current testing procedures. Tools shown in this demonstration will be released along with the talk.

Security Through Design - Making Security Better by Designing for People

In this session we will explore why certain devices, pieces of software or companies lead us to utter frustration while others consistently delight us and put a smile on our face. With these insights in mind, we will explore how we typically create our security processes, teams and solutions. All too often we create something without properly understanding what our colleagues or customers are trying to achieve only to bombard them with awareness training and policies because they "just don't get it" and because "humans are the weakest link." We will look at user-centered design methods and concepts from other disciplines like economy, psychology or marketing that can help us to build security in a truly usable way not just our tools but also the way we setup our teams, the way we communicate and the way we align incentives. Every interaction with security is an opportunity to improve convenience and bring a smile to somebody's face. By understanding the impact of design, we can do a lot to improve corporate productivity and security itself.

SGX Secure Enclaves in Practice: Security and Crypto Review

Software Guard Extensions (SGX) is a technology available in Intel(R) CPUs released in autumn 2015. SGX allows a remote server to process a client's secret data within a software enclave that hides the secrets from the operating system, hypervisor, and even BIOS or chipset manager, while giving cryptographic evidence to the client that the code has been executed correctly the very definition of secure remote computation.

This talk is the first public assessment of SGX based on real SGX-enabled hardware and on Intel's software development environment. While researchers already scrutinized Intel's partial public documentation, many properties can only be verified and documented by working with the real thing: What's really in the development environment? Which components are implemented in microcode and which are in software? How can developers create secure enclaves that won't leak secrets? Can the development environment be trusted? How to debug and analyze SGX software? What crypto schemes are used in SGX critical components? How reliable are they? How safe are their implementations? Based on these newly documented aspects, we'll assess the attack surface and real risk for SGX users. We'll then present and demo proofs-of-concept of cryptographic functionalities leveraging SGX: secure remote storage and delegation (what fully homomorphic encryption promises, but is too slow to put in practice), and reencryption. We'll see how basic architectures can deliver powerful crypto functionalities with a wide range of applications. We'll release code as well as a tool to extract and verify an enclave's metadata.

Side-Channel Attacks on Everyday Applications

In 2013, Yuval Yarom and Katrina Falkner discovered the FLUSH+RELOAD L3 cache side-channel. So far it has broken numerous implementations of cryptography including, notably, the AES and ECDSA in OpenSSL and the RSA GnuPG. Given FLUSH+RELOAD's astounding success at breaking cryptography, we're lead to wonder if it can be applied more broadly, to leak useful information out of regular applications like text editors and web browsers whose main functions are not cryptography.

In this talk, I'll briefly describe how the FLUSH+RELOAD attack works, and how it can be used to build input distinguishing attacks. In particular, I'll demonstrate how when the user Alice browses around the top 100 Wikipedia pages, the user Bob can spy on which of those pages she's visiting.

This isn't an earth-shattering attack, but as the code I'm releasing shows, it can be implemented reliably. My goal is to convince the community that side channels, FLUSH+RELOAD in particular, are useful for more than just breaking cryptography. The code I'm releasing is a starting point for developing better attacks. If you have access to a vulnerable CPU running a suitable OS, you should be able to reproduce the attack within minutes after watching the talk and downloading the code.

Subverting Apple Graphics: Practical Approaches to Remotely Gaining Root

Apple graphics, both the userland and the kernel components, are reachable from most of the sandboxed applications, including browsers, where an attack can be launched first remotely and then escalated to obtain root privileges. On OS X, the userland graphics component is running under the WindowServer process, while the kernel component includes IOKit user clients created by IOAccelerator IOService. Similar components do exist on iOS system as well. It is the counterpart of "Win32k.sys" on Windows. In the past few years, lots of interfaces have been neglected by security researchers because some of them are not explicitly defined in the sandbox profile, yet our research reveals not only that they can be opened from a restrictive sandboxed context, but several of them are not designed to be called, exposing a large attack surface to an adversary. On the other hand, due to its complexity and various factors (such as being mainly closed source), Apple graphics internals are not well documented by neither Apple nor the security community. This leads to large pieces of code not well analyzed, including large pieces of functionality behind hidden interfaces with no necessary check in place even in fundamental components. Furthermore, there are specific exploitation techniques in Apple graphics that enable you complete the full exploit chain from inside the sandbox to gain unrestricted access. We named it "graphic-style" exploitation.

In the first part of the talk, we introduce the userland Apple graphics component WindowServer. We start from an overview of WindowServer internals, its MIG interfaces as well as "hello world" sample code. After that, we explain three bugs representing three typical security flaws: - Design related logic issue CVE-2014-1314, which we used at Pwn2Own 2014 - Logic vulnerability within hidden interfaces - The memory corruption issue we used at Pwn2Own 2016 Last but not least we talk about the "graphic-style" approach to exploit a single memory corruption bug and elevate from windowserver to root context.

The second part covers the kernel attack surface. We will show vulnerabilities residing in closed-source core graphics pipeline components of all Apple graphic drivers including the newest chipsets, analyze the root cause and explain how to use our "graphic-style" exploitation technique to obtain root on OS X El Capitan at Pwn2Own 2016. This part of code, mostly related to rendering algorithm, by its nature lies deeply in driver's core stack and requires much graphical programming background to understand and audit, and is overlooked by security researchers. As it's the fundamental of Apple's rendering engine, it hasn't been changed for years and similar issues do exist in this blue ocean. We'll also come up with a new way of kernel heap spraying, with less side-effect and more controllable content than any other previous known methods. The talk is concluded by showing two live demos of remote gaining root through a chain of exploits on OS X El Capitan. Our first demo is done by exploiting userland graphics and the second by exploiting kernel graphics.

TCP Injection Attacks in the Wild - A Large Scale Study

In this work we present a massively large-scale survey of Internet traffic that studies the practice of false content injections on the web. We examined more than 1.5 Peta-bits of data from over 1.5 million distinct IP addresses. Earlier this year we have shown that false content injection is practiced by network operators for commercial

purposes. These network operators inject advertisements and malware into webpages viewed by potentially ALL users on the Internet.

In this presentation we recap the injections we discovered earlier this year and show them in detail. Additionally, we shall show new types of non-commercial injections, identify the injectors behind them and discuss their modi operandi. Finally, we shall discuss in detail analysis of a targeted injection attack against an American website.

The attacks we discovered are done using out-of-band TCP injection of false packets (rather than in-band alteration of the original packets). This is what actually allowed us to detect the injection events in the first place. We also present a novel client-side tool to mitigate such attacks that has minimal performance impact.

The Art of Defense - How Vulnerabilities Help Shape Security Features and Mitigations in Android

Information security is ever evolving, and Android's security posture is no different. Android users faces threats from a variety of sources, from the mundane to the extraordinary. Lost and stolen devices, malware attacks, rooting vulnerabilities, malicious websites, and nation state attackers are all within the Android threat model, and something the Android Security Team deals with daily. In this talk, we will cover the threats facing Android users, using both specific examples from previous Black Hat conferences and published research, as well as previously unpublished threats. For the threats, we will go into the specific technical controls which contain the vulnerability, as well as newly added Android N security features which defend against future unknown vulnerabilities. Finally, we'll discuss where we could go from here to make Android, and the entire computer industry, safer.

The Art of Reverse Engineering Flash Exploits

Adobe Flash is one of the battlegrounds of exploit and mitigation methods. As most of the Flash exploits demonstrate native memory layer exploit technique, it is valuable to understand the memory layout and behavior of Adobe Flash Player. We developed fine-grained debugging tactics to observe memory exploit technique and the way to interpret them effectively. This eventually helps defenders to understand new exploit techniques that are used for current targets quickly. This information is also valuable in deciding which area should defenders focus on for mitigation and code fixes. Adobe Flash Player was one of the major attack targets in 2015. We observed at least 17 effective zero-days or 1-day attacks in the wild. Flash is not just used by exploit kits like Angler, it has also been commonly used for advanced persistent threat (APT) attacks. The bug class ranges from simple heap overflows, uninitialized memory to type confusion and use-after-free. At Microsoft, understanding exploits in-the-wild is a continuous process. Flash exploit is one of the hardest to reverse-engineer. It often involves multi-layer

obfuscation, and by default, is highly obfuscated and has non-decompilable codes. The challenge with Flash exploit comes from the lack of tools for static and dynamic analysis. Exploits are written with ActionScript programming language and obfuscated in bytecode level using commercial-grade obfuscation tools. Understanding highly obfuscated logic and non-decompilable AVM bytecode is a big challenge. Especially, the lack of usable debuggers for Flash file itself is a huge hurdle for exploit reverse engineers. It is just like debugging PE binaries without using Windbg or Olly debugger. The ability of the researcher is highly limited.

With this presentation, I want to deliver two things: 1. The tactics and debugging technique that can be used to reverse engineer exploits. This includes using existing toolsets and combining them in an effective way. 2. The detailed exploit code reverse engineering examples that can help you understand what's the current and past status of attack and mitigation war. You might have heard of Vector corruption, ByteArray corruption and other JIT manipulation technique. Technical details will be discussed on how the exploits are using these and how the vendor defended against these.

The Beast Within - Evading Dynamic Malware Analysis Using Microsoft COM

Microsoft Common Object Model (COM) is a technology for providing a binary programming interface for Windows programs. Despite its age it still forms the internal foundation of many new Microsoft technologies such as .NET. However, over the course of more than twenty years of development, the inevitable pressure to retain backwards compatibility has turned the COM runtime into an obscure beast. These days, many COM interfaces exist that mirror almost the same functionality provided by common Windows APIs. Malware authors can easily execute almost any operation (creating files, starting new processes, etc.) only using COM calls. Dynamic malware analyzers must deal with this accordingly without getting lost in the shadowy depths of the COM runtime.

The talk presents various aspects of automated dynamic COM malware analysis and shows which approaches are actually practical and which ones are hopeless from the beginning. We show how COM interfaces are already actively used by malware in the wild. Our data retrieved from various sample sharing programs indicates that COM use is widespread and not only limited to sophisticated attacks. It can be used to create arbitrary files, access the registry, control the Windows firewall, tap into audio interfaces and much more. The possibilities are endless. Furthermore, many script engines such as VBScript or JScript use COM underneath. If such samples are analyzed, then this must be dealt with appropriately. Unfortunately, many existing dynamic analysis solutions fail at monitoring COM correctly which makes it easy for malware to evade many common sandboxes. One essential problem is that COM classes can be implemented in various places: in the calling program itself, in other processes on the same machine, or even in remote processes on different machines using DCOM. This requires to catch and process COM calls at the very first API layer and not later on. Due to the myriad of COM calls in question, hooking-based solutions quickly hit a wall. The popular workaround is to hook on API layers behind (such as NTDLL). Since COM calls can be executed in remote processes and heavily rely on data marshalling, this approach can only be used for not more than simple COM interfaces. Furthermore, it requires filtering out

irrelevant API calls from OS libraries, which, notwithstanding the above, poses many problems by itself. Last but not least, hooking COM calls (or API calls in general) makes it easy for malware to detect that it is running in a sandbox.

We show how transition-based monitoring can be used to monitor all COM calls at the first interface layer. This requires additional effort in parsing the numerous different formats COM uses to encode function call parameters. We show what obstacles are to be expected and how to deal with them accordingly. This generic approach yields a detailed list of all COM calls executed by malware with all their parameters. In addition, malware cannot evade the analysis since transitions are detected transparently in a hypervisor. Not a single bit has to be modified in the analysis environment.

The Linux Kernel Hidden Inside Windows 10

Initially known as "Project Astoria" and delivered in beta builds of Windows 10 Threshold 2 for Mobile, Microsoft implemented a full blown Linux 3.4 kernel in the core of the Windows operating system, including full support for VFS, BSD Sockets, ptrace, and a bonafide ELF loader. After a short cancellation, it's back and improved in Windows 10 Anniversary Update ("Redstone"), under the guise of Bash Shell interoperability. This new kernel and related components can run 100% native, unmodified Linux binaries, meaning that NT can now execute Linux system calls, schedule thread groups, fork processes, and access the VDSO!

As it's implemented using a full-blown, built-in, loaded-by-default, Ring 0 driver with kernel privileges, this not a mere wrapper library or user-mode system call converter like the POSIX subsystem of yore. The very thought of an alternate virtual file system layer, networking stack, memory and process management logic, and complicated ELF parser and loader in the kernel should tantalize exploit writers - why choose from the attack surface of a single kernel, when there's now two?

But it's not just about the attack surface - what effects does this have on security software? Do these frankenLinux processes show up in Procmon or other security drivers? Do they have PEBs and TEBs? Is there even an EPROCESS? And can a Windows machine, and the kernel, now be attacked by Linux/Android malware? How are Linux system calls implemented and intercepted?

As usual, we'll take a look at the internals of this entirely new paradigm shift in the Windows OS, and touch the boundaries of the undocumented and unsupported to discover interesting design flaws and abusable assumptions, which lead to a wealth of new security challenges on Windows 10 Anniversary Update ("Redstone") machines.

The Remote Malicious Butler Did It!

An Evil Maid attack is a security exploit that targets a computing device that has been left unattended. An evil maid attack is characterized by the attacker's ability to physically access the target multiple times without the owner's knowledge. On BlackHat Europe 2015, Ian Haken in his talk "Bypassing Local Windows Authentication

to Defeat Full Disk Encryption" had demonstrated a smart Evil Maid attack which allows the attacker to bypass Bitlocker disk encryption in an enterprise's domain environment. The attacker can do so by connecting the unattended computer into a rogue Domain Controller and abusing a client side authentication vulnerability. As a result, Microsoft had released a patch to fix this vulnerability and mitigate the attack. While being a clever attack, the physical access requirement for the attack seems to be prohibitive and would prevent it from being used on most APT campaigns. As a result, defenders might not correctly prioritize the importance of patching it.

In our talk, we reveal the "Remote Malicious Butler" attack, which shows how attackers can perform such an attack, remotely, to take a complete control over the remote computer. We will dive into the technical details of the attack including the rogue Domain Controller, the client-side vulnerability and the Kerberos authentication protocol network traffic that ties them. We would explore some other attack avenues, all leveraging on the rogue Domain Controller concept. We would conclude with the analysis of some practical generic detection and prevention methods against rogue Domain Controllers.

The Risk from Power Lines: How to Sniff the G3 and Prime Data and Detect the Interfere Attack

Power line communication (PLC) is a kind of communication technology which uses the power line as the communication media. The PLC technology is divided with 2 sub-field: narrow-band PLC and wide-band PLC. For the narrow-band PLC, there are 2 very import standards: Prime and G3. Both the standards are widely used in AMR and electric monitor system and it lead to the rise of threat in AMR system security and electric safety. This topic will talk about how to get the PLC data stream in a PLC communication system which would use G3 or Prime standard, and will also talk about how to detect attacking in the net. We will focus on how to identify which kind of standard the system using and how to sniff the PLC data in physical level.

The Tao of Hardware the Te of Implants

Embedded, IOT, and ICS devices tend to be things we can pick up, see, and touch. They're designed for nontechnical users who think of them as immutable hardware devices. Even software security experts, at some point, consider hardware attacks out of scope. Thankfully, even though a handful of hardware manufacturers are making some basic efforts to harden devices, there's still plenty of cheap and easy ways to subvert hardware. The leaked ANT catalog validated that these cheap hardware attacks are worthwhile. The projects of the NSA Playset have explored what's possible in terms of cheap and easy DIY hardware implants, so I've continued to apply those same techniques to more embedded devices and industrial control systems. I'll show off a handful of simple hardware implants that can 1) Blindly escalate privilege using JTAG 2) Patch kernels via direct memory access on

an embedded device without JTAG 3) Enable wireless control of the inputs and outputs of an off-the-shelf PLC 4) Hot-plug a malicious expansion module onto another PLC without even taking the system offline and 5) Subvert a system via a malicious display adapter. Some of these are new applications of previously published implants - others are brand new.
I'll conclude with some potential design decisions that could reduce vulnerability to implants, as well as ways of protecting existing hardware systems from tampering.

The Year in Flash

Adobe Flash continues to be a popular target for attackers in the wild. As an increasing number of bug fixes and mitigations are implemented, increasingly complex vulnerabilities and exploits are coming to light. This talk describes notable vulnerabilities and exploits that have been discovered in Flash in the past year.

It will start with an overview of the attack surface of Flash, and then discuss how the most common types of vulnerabilities work. It will then go through the year with regards to bugs, exploits and mitigations. It will end with a discussion of the future of Flash attacks: likely areas for new bugs, and the impact of existing mitigations.

Timing Attacks Have Never Been So Practical: Advanced Cross-Site Search Attacks

Cross-site search (XS-search) is a practical timing side-channel attack that allows the extraction of sensitive information from web-services. The attack exploits inflation techniques to efficiently distinguish between search requests that yield results and requests that do not. This work focuses on the response inflation technique that increases the size of the response; as the difference in the sizes of the responses increases, it becomes easier to distinguish between them. We begin with browser-based XS-search attack and demonstrate its use in extracting users' private data from Gmail and Facebook. The browser-based XS-search attack exploits the differences in the sizes of HTTP responses, and works even when significant inflation of the response is impossible. This part also involves algorithmic improvements compared to previous work. When there is no leakage of information via the timing side channel it is possible to use second-order (SO) XS-search, a novel type of attack that allows the attacker to significantly increase the difference in the sizes of the responses by planting maliciously crafted record into the storage. SO XS-search attacks can be used to extract sensitive information such as email content of Gmail and Yahoo! users, and search history of Bing users.

Towards a Holistic Approach in Building Intelligence to Fight Crimeware

To defeat your adversaries, it is crucial to understand how they operate and to develop a comprehensive view of their playing field. In this talk, we describe a holistic and scalable approach to investigating and combating cybercrime. Our strategy focuses on two perspectives: the network attack surface and the actors. The network attack surface exploited by malware manifests itself through various aspects such as hosting IP space, DNS traffic, open ports, BGP announcements, ASN peerings, and SSL certificates. The actors' view tracks trends, motivations, and TTPs of cyber criminals by infiltrating and maintaining access to closed underground forums where threat actors collaborate to plan cyber attacks. Crimeware campaigns nowadays rely heavily on bulletproof hosting for scalable deployment. We distinguish two types of such hosting infrastructures: the first consists of a large number of infected residential hosts scattered geographically that are leveraged to build a fast flux proxy network. This network is a hosting-as-a service platform for various malware and ransomware C2, phishing, carding, and botnet panels. The second type exists in dedicated servers acquired from rogue hosting companies or large abused hosting providers with the purpose of hosting exploit kits, phishing, malware C2, and other gray content. We start by using DNS traffic analysis and passive DNS mining algorithms to massively detect malware domains. After we identify the hosting IPs of these domains, we will demonstrate novel methods using DNS PTR data to further map out the entire IP space of bulletproof hosters serving these attacks. In the case of fast flux proxy networks, we leverage SSL data to map out larger sets of compromised hosts. Concurrently, we investigate underground forums for emerging signals about bulletproof hosters just about to be employed for malware campaigns.

The talk describes how to proactively bridge the gap between the actors and network views by identifying the IP space of the mentioned hosters given very few initial indicators and predictively block it. This is made possible thanks to the deployment at large scale of DNS PTR, SSL, and HTTP data provided by Project Sonar datasets and our own scanning of certain IP regions. It is undoubtedly a serious challenge facing security researchers to devise means to quickly index and search through vast quantities of security related log data. Therefore, we will also describe the backend architecture, based on HBase and ElasticSearch, that we use to index global Internet metadata so it is easily searchable and retrievable. Join us in this talk to learn about effective methods to investigate malware from both network and actors' perspectives and hear about our experience on how to deploy and mine large scale Internet data to support threat research.

Understanding HL7 2.x Standards Pen Testing and Defending HL7 2.x Messages

Health Level-7 or HL7 refers to a set of international standards for transfer of clinical and administrative data between software applications used by various healthcare providers. Healthcare provider organizations typically have many different computer systems used for everything from billing records to patient tracking. All of these systems should communicate with each other (or "interface") when they receive new information, or when they wish to retrieve information, but not all do so. The Hl7 2.x protocol was designed keeping certain factors in mind. Some of which are: a closed network, no malicious intent by the devices, and running the devices in a completely reliable environment. The number of devices using the HL7 2.x is huge (currently, the HL7 v2.x messaging standard is supported by every major medical information systems vendor in the world). However, a secure implementation standard / guide still needs to be worked on. Over some time I have observed that hospitals and

vendors do not fully understand the risks on their infrastructure. Also vendors need to implement some changes over their software and hardware to make their devices more resilient to attacks.

The talk will cover HL7 2.x messages, their significance and the information in these messages, also the impact of gaining access to these messages. We will look the scenario of gaining patient information, fingerprinting architecture, examining and changing diagnosis, gaining access to non-prescribed drugs / changing medication and possible financial scams. This talk will also cover how to Pen test medical systems running HL7 interfaces (EMR Software, Patient monitors, X-ray machines.. etc.), discovering common flaws and attack surfaces and on devices that use HL 7 2.x messages to test machine interfaces and connected environment.

Unleash the Infection Monkey: A Modern Alternative to Pen-Tests

Security breaches never happen exactly the way you expected or planned for. Yet an organization's infrastructure should be able to withstand a breach of its perimeter security layer, and also handle the infection of internal servers. The security testing toolset available to security professionals today consists mainly of penetration testing and vulnerability scanners.These tools were designed for traditional, relatively static networks and can no longer address ALL the possible vulnerabilities of today's dynamic and hybrid network. While there is no replacement to a highly skilled human pen-test hacker, penetration tests are limited to specific parts of a network, are expensive, and may become obsolete within months. Automatic vulnerability scanners have limited accessibility and can not simulate today's advanced lateral movement attack methods. The result is network blind spots which is where security threats often arise. This calls for a new approach to testing network security resilience. An ideal tool would be easy to use, budgetary conscious, autonomous and scalable.

We propose using the Infection Monkey, a new open source cyber security testing tool, designed to thoroughly test a network from an attacker's point of view. Our tool draws its inspiration from Netflix's Chaos Monkey released in 2011. Netflix's Monkey was designed to randomly delete servers in Netflix' infrastructure to test a service's ability to withstand server failures. We think that a similar approach applies to network security, "infecting" your network to test your defenses capabilities, so we have leveraged Netflix's Chaos Monkey concept to address the challenges of the network defense community. The Infection Monkey spins up an infected virtual machine inside random parts of your data center, to test for potential security failures. By "inside", we mean behind the firewall and any other perimeter defense you are deploying for your computing infrastructure. By equipping the monkey with advanced exploitation abilities (without destructive payloads), it can spread to any vulnerable machine within reach. Along with the ability to spread onwards from its victims, the monkey can detect surprising weak spots throughout the network.

In our talk we will show how our Infection Monkey uncovers blind spots and argue that ongoing network-wide security testing adds strong capabilities to the security team. We will focus on vulnerabilities that up until now have stayed in the industry's 'collective blind spot'. The security community can greatly benefit from a disruptive, modern tool that helps verify security solution deployments and shed light on the weaker parts of the security chain.

Using an Expanded Cyber Kill Chain Model to Increase Attack Resiliency

The Cyber Kill Chain model provides a framework for understanding how an adversary breaches the perimeter to gain access to systems on the internal network. However, this model is incomplete and can lead to over-focusing on perimeter security, to the detriment of internal security controls. In this presentation, we'll explore an expanded model including the Internal Kill Chain and the Target Manipulation Kill Chain.

We'll review what actions are taken in each phase, and what's necessary for the adversary to move from one phase to the next. We'll discuss multiple types of controls that you can implement today in your enterprise to frustrate the adversary's plan at each stage, to avoid needing to declare "game over" just because an adversary has gained access to the internal network. The primary limiting factor of the traditional Cyber Kill Chain is that it ends with Stage 7: Actions on Objectives, conveying that once the adversary reaches this stage and has access to a system on the internal network, the defending victim has already lost. In reality, there should be multiple layers of security zones on the internal network, to protect the most critical assets. The adversary often has to move through numerous additional phases in order to access and manipulate specific systems to achieve his objective. By increasing the time and effort required to move through these stages, we decrease the likelihood of the adversary causing material damage to the enterprise.

Using EMET to Disable EMET

Microsoft's Enhanced Mitigation Experience Toolkit (EMET) is a project that adds security mitigations to user mode programs beyond those built in to the operating system. It runs inside "protected" programs as a Dynamic Link Library (DLL), and makes various changes in order to make software exploitation expensive. If an attacker can bypass EMET with significantly less work, then it defeats EMET's purpose of increasing the cost of exploit development. In this briefing we discuss protections being offered from EMET, how individually each of them can be evaded by playing around the validation code and then a generic disabling method, which applies to multiple endpoint products and sandboxing agents relying on injecting their Dynamic Link Library into host processes in order to protect them. It can be noted that Microsoft has issued a patch to address this very issue in EMET 5.5 in February 2016. EMET was designed to raise the cost of exploit development and not as a "fool proof exploit mitigation solution". Consequently, it is no surprise that attackers who have read/write capabilities within the process space of a protected program can bypass EMET by systematically defeating its mitigations. As long as their address space remains same, a complete defensive solution cannot be used to prevent exploitation.

The talk will focus on how easy is it to defeat EMET or any other Agent. How secure is any endpoint exploit prevention/detection solution, which relies on same address space validations and how to defeat them with their own checks or by circumventing and evading their validation. Moreover it will also reflect on, targeted EMET evasion i.e. when the attacker knows EMET is installed on victim machine. These methods applied on EMET can be applied on other enterprise products and were tested on many during our research.

Using Undocumented CPU Behavior to See into Kernel Mode and Break KASLR in the Process

Typically, hackers focus on software bugs to find vulnerabilities in the trust model of computers. In this talk, however, we'll focus on, how the micro architectural design of computers and how they enable an attacker to breach trust boundaries. Specifically, we'll focus on how an attacker with no special privileges can gain insights into the kernel and how these insights can enable further breaches of security. We will focus on the x86-64 architecture, but round up with comments on how our research touches on ARM processors. Unlike software bugs, micro architectural design issues have applications across operating systems and are independent of easily fixable software bugs. In modern operating systems the security model is enforced by the kernel. The kernel itself runs in a processor supported and protected state often called supervisor or kernel mode. Thus the kernel itself is protected from introspection and attack by hardware. We will present a method that'll allow for fast and reliable introspection into the memory hierarchy in the kernel based on undocumented CPU behavior and show how attackers could make use of this information to mount attacks on the kernel and consequently of the entire security model of modern computers. Making a map of memory and breaking KASLR Modern operating systems use a number of methods to prevent an attacker from running unauthorized code in kernel mode. They range from requiring user-privileges to load drivers, over driver signing to hardware enabled features preventing execution in memory marked as data such as DEP (Data Execution Prevention) or more resonantly SMEP that prevents execution of user allocated code with kernel level privileges. Often used bypasses modify either page tables or use so called code reuse attacks. Either way an attacker needs to know where the code or page tables are located. To further complicate an attack modern operating system is equipped with "Kernel Address Space Randomized Layout" (KASLR) that randomizes the location of important system memory.

We'll present a fast and reliable method to map where the kernel has mapped pages in the kernel mode area. Further, we'll present a method for locating specific kernel modules thus by passing KASLR and paving the way for classic privileged elevation attacks. Neither method requires any special privileges and they even run from a sandboxed environment. Also relevant is that our methods are more flexible than traditional software information leaks, since they leak information on the entire memory hierarchy. The core idea of the work is that the prefetch instructions leaks information about the caches that are related to translating a virtual address into a physical address. Also significant is that the prefetch instruction is unprivileged and does not cause exceptions nor does it have any privilege verification. Thus it can be used on any address in the address space. Physical to virtual address conversion A number of micro-architectural attacks is possible on modern computers. The Row hammer is probably the most famous of these attacks. But attacks methodologies such as cache side channel attacks have proven to be able to exfiltrate private data, such as private keys, across trust boundaries. These two attack methodologies have in common that they require information about how virtual memory is mapped to physical memory. Both methodologies have thus far either used the "/proc/PID/pagemap" which is now accessible only with administrator privileges or by using approximations. We will discuss a method where an unprivileged user is able to reconstruct this mapping. This goes a long way towards making the row hammer attack a practical attack

vector and can be a valuable assistance in doing cache side channel attacks. Again we use the prefetch's instructions lack of privilege checking, but instead of using the timing that it leaks we now use the instructions ability to load CPU caches and that timing of memory access instructions depend heavily on the cache state. Bonus material We will shortly discuss the attack vectors relevance on ARM platforms and its potential impact on hypervisor environments. Finally, we will shortly outline a possible defense.

Viral Video - Exploiting SSRF in Video Converters

Many web applications allow users to upload video - video/image hostings, cloud storages, social networks, instant messengers, etc. Typically, developers want to convert user uploaded files into formats supported by all clients. The number of input formats is very big, so developers use third-party tools/libraries for video encoding. The most common solution in this area is ffmpeg and its forks. ffmpeg by default supports many different formats, including playlists (files with a set of links to other files). In this Briefing, we will examine exploitation of SSRF in hls (m3u8) playlists processing. Video processing is frequently done in clouds, which by design is more vulnerable to SSRF attacks, and playlists support many different protocols (http, file, tcp, upd, gopher ...), so SSRF in playlist processing can be very critical and even lead to full service takeover.

We will show how implementation details of hls playlists processing in ffmpeg allow reading files from the video conversion server, with and without network support. We will show how SSRF in video converter can give full access to service based on cloud like Amazon AWS. We will also present our tool for the detection and exploitation of this vulnerability. We will show a truly "viral" video which could perform successful attacks on Facebook, Telegram, Microsoft Azure, flickr, one of Twitter services, Imgur and others.

VOIP WARS: The Phreakers Awaken

Larger organisations are using VoIP within their commercial services and corporate communications and the take up of cloud based Unified Communications (UC) solutions is rising every day. However, response teams and security testers have limited knowledge of VoIP attack surfaces and threats in the wild. Due to this lack of understanding of modern UC security requirements, numerous service providers, larger organisations and subscribers are leaving themselves susceptible to attack. Current threat actors are repurposing this exposed infrastructure for botnets, toll fraud etc.

The talk aims to arm response and security testing teams with knowledge of cutting-edge attacks, tools and vulnerabilities for VoIP networks. Some of the headlines are: attacking cloud based VoIP solutions to jailbreak tenant environments; discovering critical security vulnerabilities with the VoIP products of major vendors; exploiting harder to fix VoIP protocol and service vulnerabilities; testing the security of IP Multimedia Subsystem

(IMS) services; and understanding the toolset developed by the author to discover previously unknown vulnerabilities and to develop custom attacks. In addition, the business impact of these attacks will be explained for various implementations, such as cloud UC services, commercial services, service provider networks and corporate communication. Through the demonstrations, the audience will understand how can they secure and test their communication infrastructure and services. The talk will also be accompanied by the newer versions of Viproy and Viproxy developed by the author to operate the attack demonstrations.

Watching Commodity Malware Get Sold to a Targeted Actor

Detected breaches are often classified by security operation centers and incident response teams as either "targeted" or "untargeted." This quick classification of a breach as "untargeted," and the following de-prioritization for remediation, often misses a re-classification and upgrade process several attack groups have been conducting. As part of this process, assets compromised as part of broad, untargeted "commodity" malware campaigns are re-classified based on the organizational network they're part of to determine their potential value in the market. The higher value ones are upgraded and taken out of the "commodity" campaign to prepare them for a sale, for buyers planning a targeted attack. Organizations overlooking this often miss the opportunity to eliminate the threat prior to its escalation.

This session will cover the analysis of endpoint and network data captured during these re-classification operations, demonstrating the techniques and procedures used by some of the attack groups as they migrate compromised endpoints from the "commodity" threat platform to the valuable-target's platform. What measures can be taken to detect that a commodity threat is going through a migration process? How can this be leveraged to increase the efficiency of the incident response process?

Weaponizing Data Science for Social Engineering: Automated E2E Spear Phishing on Twitter

Historically, machine learning for information security has prioritized defense: think intrusion detection systems, malware classification and botnet traffic identification. Offense can benefit from data just as well. Social networks, especially Twitter with its access to extensive personal data, bot-friendly API, colloquial syntax and prevalence of shortened links, are the perfect venues for spreading machine-generated malicious content.

We present a recurrent neural network that learns to tweet phishing posts targeting specific users. The model is trained using spear phishing pen-testing data, and in order to make a click-through more likely, it is dynamically seeded with topics extracted from timeline posts of both the target and the users they retweet or follow. We augment the model with clustering to identify high value targets based on their level of social engagement such as their number of followers and retweets, and measure success using click-rates of IP-tracked links. Taken together, these techniques enable the world's first automated end-to-end spear phishing campaign generator for Twitter.

Web Application Firewalls: Analysis of Detection Logic

The presentation will highlight the core of Web Application Firewall (WAF): detection logic, with an accent on regular expressions detection mechanism. The security of 6 trending opensource WAFs (OWASP CRS 2,3 - ModSecurity, Comodo WAF, PHPIDS, QuickDefense, Libinjection) will be called into question.

Static Application Security Testing (SAST) tool for Regular Expressions analysis will be released, which aims to finds security flaws in the cunning syntax of regular expressions. Using the proposed "regex security cheatsheet", rules from popular WAFs will be examined. Logical flaws in regular expressions will be demonstrated by applying author's bughunting experience and best practices. Unexpected by regexp's primary logic vectors will be discovered for Cross-Site Scripting and SQL-Injection attacks (MySQL, MSSQL, Oracle) using advanced fuzz testing techniques. Obtained from fuzz testing framework attack vectors will be clustered and represented via look-up tables. Such tables can be used by both attackers and defenders in order to understand the purpose of characters in various parts of attack vector, which are allowed by appropriate browsers or databases.

More than 15 new bypass vectors will be described, with an indication of over 300 potential weakness in regular expression detection logic of WAFs.

What's the DFIRence for ICS?

Digital Forensics and Incident Response (DFIR) for IT systems has been around quite a while, but what about Industrial Control Systems (ICS)? This talk will explore the basics of DFIR for embedded devices used in critical infrastructure such as Programmable Logic Controllers (PLCs), Remote Terminal Units (RTUs), and controllers. If these are compromised or even have a misoperation, we will show what files, firmware, memory dumps, physical conditions, and other data can be analyzed in embedded systems to determine the root cause.

This talk will show examples of what and how to collect forensics data from two popular RTUs that are used in Electric Substations: the General Electric D20MX and the Schweitzer Engineering Labs SEL-3530 RTAC.

This talk will not cover Windows or *nixbased devices such as Human Machine Interfaces (HMIs) or gateways.

When Governments Attack: State Sponsored Malware Attacks Against Activists Lawyers and Journalists

Targeted malware campaigns against Activists, Lawyers and journalists are becoming extremely commonplace. These attacks range in sophistication from simple spear-phishing campaigns using off the shelf malware, to APT-level attacks employing exploits, large budgets, and increasingly sophisticated techniques. Activists, lawyers and journalists are, for the most part, completely unprepared to deal with cyber-attacks; most of them don't even have a single security professional on staff. In this session Eva
Galperin and Cooper Quintin of the Electronic Frontier Foundation will discuss the technical and operational details of malware campaigns against activists, journalists, and lawyers around the world, including EFF. They will also present brand new research about a threat actor targeting lawyers and activists in Europe and the Post-Soviet States. With targeted malware campaigns, governments have a powerful tool to suppress and silence dissent. As security professionals we are in a unique position to help in this fight.

When the Cops Come A-Knocking: Handling Technical Assistance Demands from Law Enforcement

What kind of surveillance assistance can the U.S. government force companies to provide? This issue has entered the public consciousness due to the FBI's demand in February that Apple write software to help it access the San Bernardino shooter's encrypted iPhone. Technical assistance orders can go beyond the usual government requests for user data, requiring a company to actively participate in the government's monitoring of the targeted user(s). Companies that take seriously the task of securing of their users' information and communications must be prepared to respond to demands to disclose, proactively begin storing, or decrypt user data; write custom code; allow the installation of government equipment on their systems; or hand over encryption keys. Advance preparation for handling technical assistance demands is especially important now since the U.S. Department of Justice has been so aggressive with companies that resist broad or novel surveillance orders. In the "Apple vs. FBI" case, America's richest company faced a motion for contempt of court and derisive rhetoric from U.S. officials before it enlisted the nation's top lawyers in its defense and ultimately fought off the case. In stark contrast, encrypted e-mail provider Lavabit unsuccessfully opposed multiple court orders to compel it to decrypt and give law enforcement the e-mails of its most famous customer, Edward Snowden, and even to hand over its private encryption keys. The Fourth Circuit Court of Appeal did not look kindly on Lavabit, which lost its legal battle and shuttered its operations after its legal defeat. In 2007, Yahoo! unsuccessfully battled warrantless wiretapping in secret before the Foreign Intelligence Surveillance Court. The price for seeking to protect its users' Fourth Amendment rights? DOJ argued that Yahoo! should be fined $250,000 a day for non-compliance while the litigation was pending.

This talk, given by two Crypto Policy Project attorneys from Stanford Law School's Center for Internet and Society, will teach an enterprise audience what they need to know about technical-assistance orders by U.S. law

enforcement, so that they can handle demands effectively even if they do not have Apple-level resources. We'll go over what sorts of assistance law enforcement may demand you provide (and has demanded of companies in the past), whether they have authority to require such assistance and under what law(s), and a company's options in response.

Windows 10 Mitigation Improvements

Continuous improvements have been made to Windows and other Microsoft products over the past decade that have made it more difficult and costly to exploit software vulnerabilities. The various mitigation technologies that have been created as a result have played a key role in helping to keep people safe online even as the number of vulnerabilities that are found and fixed each year has increased. In this presentation, we'll describe some of the new ways that Microsoft is tackling software security and some of the new mitigation improvements that have been made to Windows 10 as a result. This talk will cover a new data driven approach to software security at Microsoft. This approach involves proactive monitoring and analysis of exploits found in-the-wild to better understand the types of vulnerabilities that are being exploited and exploitation techniques being used. This category of analysis and insight has driven a series of mitigation improvements that has broken widely used exploitation techniques and in some cases virtually eliminated entire classes of vulnerabilities.

In this presentation, we'll share more details on how this analysis is performed at Microsoft, how it has helped drive improvements, and how we have measured the success of those improvements. This presentation will also describe Microsoft's unique proactive approach to software security assurance which embraces offensive security research and extends traditional "red team" operations into the software security world. This approach replaces traditional software security design and implementation reviews with a true end-to-end simulation of attacks in the wild by spanning vulnerability discovery, exploit development, and mitigation bypass identification. This approach enables Microsoft to concretely evaluate the effectiveness of mitigations, identify gaps in protection, and provide concrete metrics on the cost and resources required to develop an exploit in a given scenario. In other words, this provides concrete data to help Microsoft be proactive about making holistic platform security improvements rather than simply waiting and reacting to what we see attackers do in-the-wild. In order to help drive these points home, this presentation will describe a number of mitigation improvements that have been made in Windows 10 and the upcoming Windows 10 anniversary edition. We will show how these improvements were supported by the above methods and what impact we expect these improvements to have going forward. This portion of the presentation can be seen as a follow-on to our "Exploit Mitigation Improvements in Windows 8" presentation which was given at Black Hat USA 2012.

Windows 10 Segment Heap Internals

Introduced in Windows 10, Segment Heap is the native heap used in Windows app (formerly called Modern/Metro app) processes and certain system processes. This heap is an addition to the well-researched and widely documented NT heap that is still used in traditional application processes and in certain types of allocations in Windows app processes.

One important aspect of the Segment Heap is that it is enabled for Microsoft Edge which means that components/dependencies running in Edge that do not use a custom heap manager will use the Segment Heap. Therefore, reliably exploiting memory corruption vulnerabilities in these Edge components/dependencies would require some level of understanding of the Segment Heap.

In this presentation, I'll discuss the data structures, algorithms and security mechanisms of the Segment Heap. Knowledge of the Segment Heap is also applied by discussing and demonstrating how a memory corruption vulnerability in the Microsoft WinRT PDF library (CVE-2016-0117) is used to create a reliable write primitive in the context of the Edge content process.

Xenpwn: Breaking Paravirtualized Devices

Instead of simply emulating old and slow hardware, modern hypervisors use paravirtualized devices to provide guests access to virtual hardware. Bugs in the privileged backend components can allow an attacker to break out of a guest, making them quite an interesting target.

In this talk, I'll present the results of my research on the security of these backend components and discuss Xenpwn, a hypervisor based memory access tracing tool used to discover multiple critical vulnerabilities in paravirtualized drivers of the Xen hypervisor.
If you like virtualization security, race conditions, vulnerabilities introduced by compiler optimizations or are a big fan of Bochspwn, this is the right talk for you.