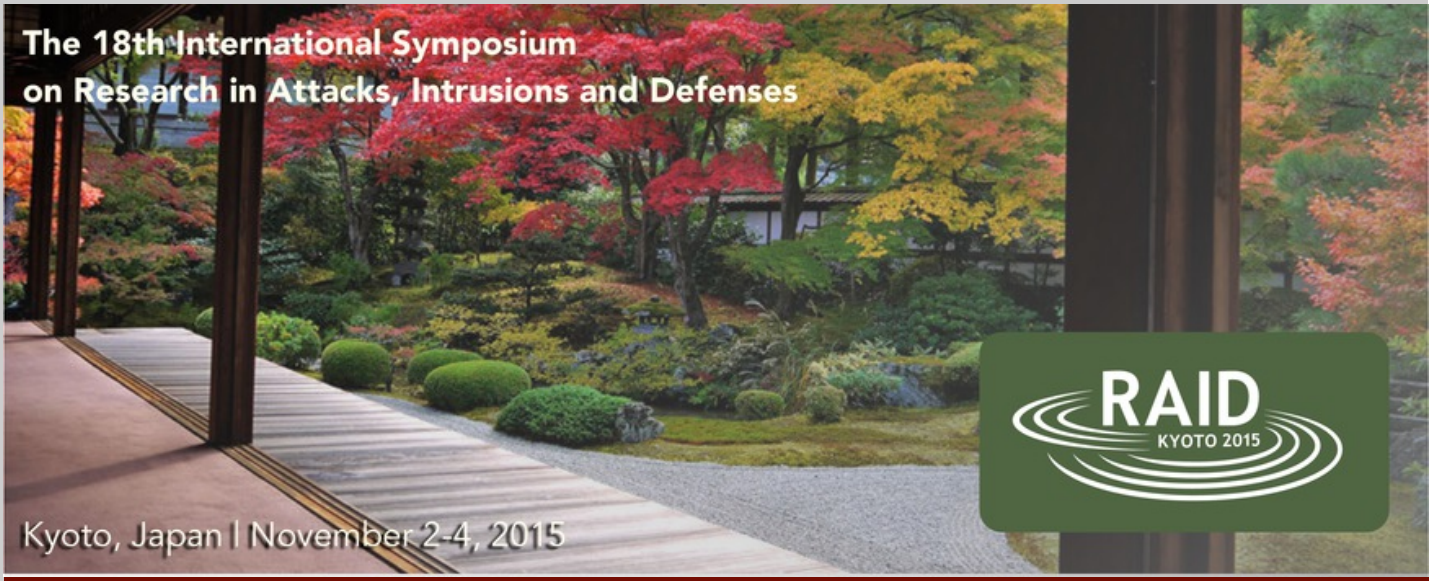

Public Review

Session 1: Cool Stuff You Can Do With Hardware

Ensemble Learning for Low-level Hardware-supported Malware Detection
Khaled N. Khasawneh; Meltem Ozsoy; Caleb Donovick; Nael Abu-Ghazaleh; Dmitry V. Ponomarev

Anti-virus software causes performance overhead on the hosts it aims to protect. To reduce this overhead, the paper investigates the effectiveness of hardware-supported malware detection. The authors consider several low-level features, which can be collected in hardware, and evaluate how well they work for particular categories of malware. They also propose metrics to quantify the benefit to anti-virus products from these hardware signals, and briefly outline a hardware implementation of the feature collection and classification system, by extending the AO486 open core and synthesizing it on an FPGA platform.

The significance of the work lies in its contribution to our understanding of hardware-based malware detection. While the security community has devoted considerable efforts to the development of software for detecting malicious programs, hardware detectors are a more recent idea. Demme et al. explored malware detection using the performance counters available on modern CPUs, and the authors' prior work proposed additional features that could potentially be collected in hardware, such as features derived from the instruction mix and memory reference patterns. Given that these low-level features are less insightful than the semantic features used in software-based detectors, a key question is how much they raise the bar for the attacker.

The program committee liked the paper's attempt to answer this question, by defining metrics to quantify the benefit that software-based anti-virus products can derive from the low-level hardware features. Specifically, the authors envision a system where the hardware detector is always on and produces a list of suspected processes, and the software detector uses this list to prioritize the usage of more heavy-weight protection mechanisms (e.g. Control Flow Integrity). The authors define and evaluate the work advantage (the ratio of the malware volumes detected with and without hardware support, in a scenario where the software detector scans only a fraction of the programs executed) and the time-to-detection (the expected time to detect a specific malware sample, when the software detector scans all the programs).

The reviewers also liked the fact that the authors break down malware into several categories (with different behaviors) and train logistic regression classifiers that are specialized to detect malware in each category. The final hardware detector is an ensemble classifier, which combines the outputs of the specialized detectors; the feature collection, the logistic regression and the combination of specialized models are implemented in hardware. The experimental results suggest that the specialized ensemble classifier has a work advantage of 11x (1.87x better than the best single detector) and reduces the time-to-detection by 6.6x when the fraction of malware programs is low (the hardware detectors exhibit diminishing returns with an increasing amount of malware). The program committee felt that these results represent a promising, but not definitive, evaluation of hardware support for anti virus scanning, as they are derived from micro-benchmarks. For example, the testing set for a specialized detector (and the experiments comparing it to the general detectors) only includes normal programs and the specialized detector's type of malware; it is less clear what happens when a certain specialized detector (e.g. for backdoors) is presented with a malware sample from another category (e.g. a worm).

In consequence, the effectiveness of hardware-based malware detection with more representative workload mixes remains an open question. In particular, a machine typically runs multiple processes, and it is unclear how easy it would be for the hardware detector to separate the features belonging to each process. Another open question is why these low level features work, and how easy it would be for an attacker to bypass them. The authors provide an intuition for this in Section 3, but a more rigorous answer to this question will require follow-on research.

Physical-layer Detection of Hardware Keyloggers
Ryan M. Gerdes; Saptarshi Mallick

This paper presents techniques to detect passive keyloggers on PS2 keyboards (and argues about USB). Detection of the keyloggers is based on the observer effect: the keylogger changes the electrical properties of the cable, which can then be detected.

The paper was found interesting and very well executed. In particular, it contains a good modelisation of the problem and an extensive evaluation (on 25 keyboards). Machine learning is used to detect the keylogger. Several scenarios are explored and evaluated, like the effect of temperature and the possibility of a high impedance keylogger. One question left is how this countermeasure could be deployed in practice.

Reverse Engineering Intel Last-Level Cache Complex Addressing Using Performance Counters
Clémentine Maurice; Nicolas Le Scouarnec; Christoph Neumann; Olivier Heen; Aurélien Francillon

This paper proposes a new methodology for reverse-engineering Intel complex addressing mechanism, a critical step for the side-channel attack on the LLC of recent processors. Such an attack requires the knowledge about how the slice is mapped to a cache-line access, which is not public and hidden by the complexity of Intel's hash-based mapping mechanism. This issue has been addressed by the authors through monitoring accesses to the LLC using the CPU’s performance counters. The authors evaluated the effectiveness of the technique across a diverse set of Intel processors.

The reviewers all agree that the new technique presented in the paper is interesting. The idea of tracking the LLC lookup counters in the performance unit through continuously addressing the same addresses is cute. Although the PC expressed some concerns about amount of manual effort that may be required for the analysis, the authors do show fully automated reversing and an end-to-end covert channel attack. All in all, the reviewers appreciated the progress made by the research, which indicates that despite such hashing-based complex addressing, the threat of the side-channel attack on LLC is still real.

Hardware-Assisted Fine-Grained Code-Reuse Attack Detection
Pinghai Yuan; Xuhua Ding; Qingkai Zeng

This paper presents the implementation of a control-flow integrity mechanism, called CFIGuard, which detects return-oriented programming attacks based on monitoring indirect branches based on Intel's last branch record (LBR) and the performance monitoring unit (PMU). The monitored branches are evaluated based on a statically derived control-flow graph (CFG). CFIGuard has been evaluated with regards to file size increase and performance overhead based on Apache, MySQL, vsftp.

CFIGuard nicely exploits the last branch record of recent Intel CPUs to validate indirect branch instructions. Since the LBR can only hold the last 16 branches, CFIGuard maintains a counter in the performance monitoring unit (PMU) to prevent flushing of the LBR buffer. Coupling the PMU with LBR allows CFIGuard to validate all indirect branches of the target user application. In contrast, previous work could be bypassed by means of the so-called LBR flushing attacks.

While this new approach demonstrates that LBR and PMU provide a solid method to validate indirect branches, there are several subtle aspects that need to be addressed in future work. First, it remains to be shown how effective and efficient CFIGuard can be applied to multiple concurrently running applications as CFI validation is performed on each context switch. Second, since LBR and PMU are originally designed for debugging purposes, we need to show that LBR-based mitigation techniques to not violate the original purpose of these technologies. Third, the frequency of CFI validation is critical and requires more investigation. Traditional CFI schemes validate indirect branches before they execute, whereas LBR-based mitigation techniques perform an CFI check after a certain window of executed indirect branches. This could potentially allow an attacker to perform the entire attack before CFI validation is performed.

Session 2: Networks

Haetae: Scaling the Performance of Network Intrusion Detection with Many-core Processors
Jaehyun Nam; Muhammad Jamshed; Byungkwon Choi; Dongsu Han; KyoungSoo Park

Haetae is a high-performance network-based intrusion detection system (NIDS) that on runs on many-core processors. In particular, the paper shows how many-core processors can be used to accelerate NIDS. While many-core processors have been used for similar purposes in previous work, this paper proposes some optimizations to reach better performance, such as shared-nothing architecture, page-locking and batching for improving PCIe throughput, and opportunistic offloading.

The engineering aspects of the paper were greatly appreciated by the PC. Haetae is able to demonstrate the improvements that are necessary in order to make the most of many-core systems and achieve the throughput needed for moderns networks. Haetae also makes use of the host CPU in order to increase the maximum achievable throughput. The approach shows promise to be ported to other programmable network interface cards (NICs) and many-core processors as well, making it more deployable.

Overall, this framework relies less on advances in security-specific technology and more on generic advances in parallel and high-speed computing. While the system uses some optimizations that are specific to the particular many-core system that is being used, the PC noted that it would be good to strive for wider deployability and identify the opportunities for a more generalized version of the work, or for the implementation of an extra abstraction layer that would remove the need for a custom implementation for every device.

Demystifying the IP Blackspace
Quentin Jacquemart; Pierre-Antoine Vervier; Guillaume Urvoy-Keller; Ernst Biersack

This paper looks at IP address blocks (CIDRs) that have not been delegated by any

authority (IANA, RIR, etc), but are being advertised via the BGP protocol, meaning that they can be used. This IP address space is referred to as the blackspace. By monitoring the RIR records, following new delegations and combining BGP announcements, the authors are able to define a methodology to identify the IP blackspace. Utilizing this method, the authors probe these networks to identify potential malicious activity, which they verify using IP blacklists and domain name mappings within that IP space.

The PC found this research very interesting as it sheds further light into this interesting phenomenon, what information is hiding in the blackspace and the kind of activities that take place there. The PC agreed that even though similar work has been done in this area and some aspects of the IP blackspace estimation and service identification could be more accurate this well-written paper provides further insights and interesting case studies which makes it a good contribution to the RAID'15 program.

Providing Dynamic Control to Passive Network Security Monitoring
Johanna Amann; Robin Sommer

This paper presents the design and implementation of a system that aims at connecting IDSes to the network control/forwarding plane, so that rules can be enforced based on IDS detections. To do so, the system exposes the capabilities of forwarding plane components to the IDS systems via a uniform API defined by the authors. This API can be used to implement low/high-level rules that can be issued by the IDS system and enforced by the forwarding elements, as a reaction to IDS detection.

As the paper acknowledges, the concept of driving the integration between IDS monitors and active network components is not new. However, this systems is a particularly flexible, elegant, and practical approach to the problem. One limitation that could be addressed in future work concerned the adaption of the technique to closed-source systems. Would the correct semantics be available to invoke the enforcement policy events?

The insights from the practical implementation and real-world deployment of a similar system are valuable, and we expect the paper and the accompanying open-source implementation will be of great interest to the RAID community.

Session 3: Hardening

Probabilistic Inference on Integrity for Access Behavior Based Malware Detection
Weixuan Mao; Zhongmin Cai; Xiaohong Guan; Don Towsley

This paper proposes a probabilistic model for detecting malware in desktop operating systems such as MS Windows. The model is based in Biba's no read down, no write up policy, but it does not fix the number of integrity levels. Instead, integrity levels are automatically learned, and the detection mechanism looks at fractions of violations for a process. The approach is evaluated on 1-2 weeks worth of data from 8 users, as well as 7,257 malware samples run in a VM without a network connection. The experiments show that the fraction of violations cannot alone be used to differentiate the benign and malicious traces; however, if the traces are processed by a Random Forests classifier, the detector can achieve 99.98% TPR with a 1.0% FPR.

The reviewers in the PC appreciated the statistical analysis presented in the paper. The proposed probabilistic integrity model appears to be very accurate. Unlike traditional integrity models the framework defines system subject/object integrity levels as dynamic, context-aware random variables. Hence, the integrity levels better reflect the time-varying trustworthiness of the entities at runtime. Additionally, the framework builds on the NRD/NWU policies with runtime tracking and propagation of integrity assessment (belief) following object access operations of subjects. Finally, it improves the accuracy of malware detection techniques based on processes' object access

behaviors with very high accuracy.

The PC reviewers had some concerns around the evaluation and the limitations of the proposed methods. It should be noted, however, that the authors discussed and clarified many of the questions raised about the evaluation in the final version of the paper.

Counteracting Data-Only Malware with Code Pointer Examination
Thomas Kittel; Sebastian Vogl; Julian Kirsch; Claudia Eckert

Code Pointer Examination (CPE) is a technique for detecting malware in Kernel code pointers. To achieve this functionality, CPE first identifies kernel code pointers and then determines if they are benign or malicious by examining if a code pointer points to legitimate targets (e.g., points to the start of a function, or is used in kernel run-time patching, etc.,)

The PC enjoyed reading this paper and appreciated the detailed work that went into making CPE robust against corner cases. In future work, it would be great to extend CPE to cover code reuse attacks, but the PC acknowledges that the current implementation raises the bar for attackers. Finally, a several reviewers raised the point that more extensive evaluation (e.g., using benchmarks that focus on exercising kernel-level code) would help readers better understand CPE's performance characteristics.

Xede: Practical Exploit Early Detection
Meining Nie; Purui Su; Qi Li; Zhi Wang; Lingyun Ying; Jinlong Hu; Dengguo Feng

This paper presents Xede, a novel approach for the detection of document exploits. The approach leverages a code emulator (Qemu) and builds a set of heuristics for the detection of common indicators of an ongoing exploit. Specifically, the system has three components that focus on different aspects of the exploitation process: a code injection detector, an exception analysis component, and a code reuse (ROP) detector. The authors describe the practical challenges of building such as system and also provide a detailed evaluation of the detection effectiveness and performance.

The PC concluded that the amount of work and the "density" of the work described in the paper was impressive. Specifically, the reviewers felt that the authors built a real system that had to face the various idiosyncrasies of the Windows platform and popular applications. Moreover, the PC appreciated the fact that the authors not only ran against a large set of known exploit documents, but also deployed their system in the wild and compared the detection performance with well-known AV software.

One concern raised by the reviewers was a potential lack of novelty of the three presented detectors. However, there was a common belief that the additional heuristics and detailed descriptions of the corner cases were sufficiently insightful.

Finally, all reviewers complained about the lack of a thorough false positive evaluation in the original submission. Specifically, while the system was deployed on a Chinese University mail server and offered publicly as a web service, there was no clear idea provided on the number of interesting but benign documents that have been analyzed (e.g., benign PDF documents containing scripts). Since the authors had the information from their experiments available, this shortcoming could be properly addressed through shepherding. As a result, the paper that appears at the conference provides proper insights into possible false positives.

Session 4: Attack Detection I

Preventing Exploits in Microsoft Office Documents through Content Randomization
Charles Smutz; Angelos Stavrou

The paper presents a way to protect users from malicious (Office) documents. The idea

is a new tweak on randomization. Many exploits are included in (malicious) documents (Microsoft Office files, PDFs, etc.) and are triggered once the user is viewing the document. This paper proposes randomizing malicious documents for preventing any exploits they include from running. The idea is to shuffle the data of the documents, while preserving the semantics of the document's format and content. This shuffling may break the functionality of exploits that are encapsulated in a malicious document, and, in case the document is not malicious, the user experience is not affected, since semantics are preserved. The authors propose two techniques for randomizing the content of the document. DCFR, which acts pretty much like ASLR, the order of the data blocks is randomized, and DCER, which is based on randomizing the document's encoding. The randomization of the document should take place between creation time and consumption time. For example, documents that are sent through e-mail could be randomized before leaving the mail server. The ideas presented in this paper are applied to Microsoft Office 2003/2007 documents but they essentially can be applied to other formats as well.

This paper was well-received by PC. The idea is based on simplicity and thus it is considered attractive. Some members of the PC raised remarks about the evaluation of the system in order to justify that legitimate documents preserve their actual semantics after randomization has been applied, and how the selection of the documents that compose the evaluation sample were selected. Also, there was an important comment from a PC member who argued that the technique is not effective in preventing a shellcode from running, but rather makes it harder for the attacker to implement the shellcode, while the randomization is in place. For example an advanced exploit could be triggered before file loading and potentially include the needed functionality for handling a randomized document. However, since these sophisticated exploits have not been observed so far, the PC agreed that this work represents a move in the right direction, and handling such advanced exploitation techniques can be part of future work.

Improving accuracy of static integer overflow detection in binary
Yang Zhang; Xiaoshan Sun; Yi Deng; Liang Cheng; Shuke Zeng; Yu Fu; Dengguo Feng

INDIO is a tool that automatically detects integer overflow errors. It is composed of two components, a static taint analyzer and a symbolic execution engine. The first component identifies potential integer overflow paths, while the seconds validates whether the identified paths contain an integer overflow with the S2E symbolic execution engine. The paper also introduces vulnerability ranking to determine the vulnerabilities to validate first. The evaluation shows the effectiveness of INDIO by identifying two previously unseen errors and producing a reduced number of false reports compared with IntScope.

The PC thought that the paper describes a set of interesting techniques to optimize finding integer overflows, a serious problem that is particularly challenging on binaries. Also, the evaluation of INDIO showed positive results of the proposed techniques by identifying new vulnerabilities. The PC also noted that a more thorough evaluation with additional applications would be great. Nevertheless, it was generally felt that the paper presented a clear improvement in the area of integer-overflow detection and should be part of the program.

A Formal Framework for Program Anomaly Detection
Xiaokui Shu; Danfeng (Daphne) Yao; Barbara G. Ryder

This paper presents a formalization of program anomaly detection systems, classifies significant examples from the literature in terms of this formalization, and suggests avenues for bridging the gap between the current state-of-the-art and the theoretical limit of each level in the model's hierarchy.

The PC particularly appreciated the ability to classify any existing program anomaly detection into a formal framework. Moreover, the PC considered that the paper

successfully established strong connections between formal languages, host-based anomaly detection systems, and the expressive power of these systems, and exemplified them in terms of prior work. There was also value in identifying potential avenues of future work, although less space was devoted to exploring these.

Finally, the PC had some concerns with the fact that the formalization focuses solely on control flow and ignores data, which could be the focus of future work.

Session 5: Web and Net

jAEk: Using Dynamic Analysis to Crawl and Test Modern Web Applications
Giancarlo Pellegrino; Constantin Tschürtz; Eric Bodden; Christian Rossow

Websites are becoming increasingly complex and dynamic. While the dynamic nature of websites provides a valuable user experience, it raises technical challenges for automated tools that crawl websites for various purposes, including quality assurance and vulnerability assessment. jAEk addresses this challenge by combining traditional Web crawling techniques with client-side dynamic program analysis of HTML and JavaScript that is agnostic to the JavaScript engine. Experiments show jAEk can crawl 86% more Web application surface than prior approaches.

The PC acknowledged the challenges of, and need for, crawling dynamic Web applications. Overall, it was felt that jAEk provides a significant improvement over the state-of-the-art. It provides practical value for both industry and other researchers, particularly as an open source project. While jAEk is clearly an improvement over prior crawlers, the PC felt that the empirical comparison could have been expanded to include additional popular crawlers such as Scrapy, Apache Nutch, BurpSuite, Zed Attack Proxy (ZAP), Heritrix, Nikto, and Crawler4j.

WYSISNWIV: What You Scan Is Not What I Visit
Qilang Yang; Dimitrios Damopoulos; Georgios Portokalidis

This paper identifies discrepancies between how URLs are parsed by browsers and URL scanners as an issue for detecting malicious web pages. In particular, corner cases when parsing URLs can lead to missed detections that expose users to attacks, or false positives that reduce the effectiveness of the URL scanning services.

The reviewers agreed that the paper highlights an interesting practical problem in the context of defending users from drive-by download attacks. Reviewers also noted that the experiments considered a variety of real browsers and scanners, cementing the practicality of the work.

That said, the reviewers also felt that the scientific contribution of the paper could be strengthened. For instance, the basic problem of implementation differences between defenses and the systems they try to protect is well known. Other areas for improvement suggested include scanning for instances of evasion in the wild and examining discrepancies in other Web technologies (e.g., JavaScript). Nevertheless, the reviewers leaned positively in the end on the basis of the practicality and potential security benefit for users in documenting these issues.

SDN Rootkits: Subverting Network Operating Systems of Software-Defined Networks
Christian Röpke; Thorsten Holz

This paper investigates the feasibility of rootkit attacks on SDN controllers. In particular, the authors demonstrate that their proposed SDN rootkit could be installed on two very popular SDN controllers - in the form of a malicious SDN application. Once installed, the SDN rootkit can be used to issue commands to control the network. Three such example commands are implemented to demonstrate the feasibility of the attack.

The PC felt that the work presents an interesting problem that will become increasingly important as SDN deployments become more common. The paper makes some important points about the vulnerabilities in SDN controllers, which can lead to serious security problems as demonstrated by the presented prototype. As a preliminary step in this direction this paper leaves some open questions and room for investigation as part of future work. In particular, the PC felt it would be interested to see the threat model and the TCB become more concretely defined as we gain better understanding of the SDN vulnerability space. The prototype presented by the authors provides some very important security insights about two very popular SDN platforms, that would be undoubtedly useful to the respective development communities. In future work, it would be interesting to see the generalization of these approaches.

Finally, even though the prototype rootkit implementation does leave some room for improvement in terms of sophistication (e.g., flow discrepancies can make the rootkit easy to detect, external detection mechanisms can easily detect the presence of the rootkit, etc.), it is successful in demonstrating the existence of a real-world security problem for SDN controllers, that will need to be pursued further by researchers as well as SDN controller developers. As such, the paper is of practical importance and makes a great contribution to the program.

Session 6: Android

AppSpear: Bytecode Decrypting and DEX Reassembling for Packed Android Malware
Wenbo Yang; Juanru Li; Bodong Li; Junliang Shu; Wenjun Hu; Yuanyuan Zhang; Dawu Gu

In this paper AppSpear is presented, a framework for extracting packed code from Android applications. The authors discuss how packing Android applications is becoming increasingly common for both benign and malicious purposes. While the Android static analysis tools are getting better, packing limits what can be done. AppSpear can assist by extracting the unpacked DEX code and by creating an APK file that can be analyzed by existing tools. The authors motivate AppSpear by studying a dataset of 37,668 malware samples across 3+ years and showing that there is a growing trend towards using packers. From these packed malware apps, the authors find six prominent packers, which are also used to protect the intellectual property of legitimate apps. The AppSpear solution extracts DEX code from memory and recreates an APK for use by other tools.

The PC found AppSpear as a practical tool that is useful for the community. While the PC had some concerns regarding evasion of AppSpear and about limitations due to the inherent problems of dynamic analysis. These concerns are well-known weaknesses of all such systems and the authors address them explicitly in a discussion section which was sufficient to convince the PC that this is a nice contribution to the RAID program.

HelDroid: Dissecting and Detecting Android Ransomware
Federico Maggi; Nicolò Andronio; Stefano Zanero

This paper offers a first look at ransomware in the mobile space. In particular, it presents a new approach for detecting Android ransomware. The authors focus on a few prominent features of this type of malware, including encryption, device locking and threatening text, and utilize both static, dynamic analysis and natural language processing to identify suspicious code. The new system implemented, called Heldroid, was evaluated against two ransomware datasets constructed using VirusTotal as ground truth, and found to work effectively in detecting the malicious applications.

The PC agreed that the individual techniques were already known, but their combination in the mobile space for this problem was not previously attempted. Accordingly, the contributions were largely the observations and measurements made by the authors. Additionally, some reviewers were concerned about the performance cost of dynamic analysis on real mobile devices (i.e., practicality of deployment) and the

ability to evade the presented techniques. The authors were explicitly asked to address the issue of evasion more thoroughly. Overall, the PC found that this work conducted a novel investigation into an important space, and believes that the work will be beneficial in starting discussions.

Continuous Authentication on Mobile Devices Using Power Consumption, Touch Gestures and Physical Movement of Users
Rahul Murmuria; Angelos Stavrou; Daniel Barbara; Dan Fleck

This paper proposes a continuous authentication scheme that utilizes power consumption, touch gestures, and physical movements, arguing that different users will be easily distinguishable by this metric. These features were used to train a machine-learning classifier to build models for different applications and users. The approach is evaluated by recording the activities of 73 volunteers.

The PC agreed that the problem of authenticating users to mobile devices is an important challenge for the security community. While continuous authentication has been thoroughly investigated in recent years, this paper presents novel work in the area by combining three features. It also noted that the paper would benefit from additional information on feature selection and a more thorough evaluation with additional applications. To conclude, despite some concerns on the simplicity of the model, the PC thought that paper contains novel elements that should be included in the program.

Session 7: Privacy

Privacy Risk Assessment on Online Photos
Haitao Xu; Haining Wang; Angelos Stavrou

This paper presents a study of metadata found in photos collected from the web. The authors investigate the mechanisms which store metadata initially, how much is stored, and how much remains accessible to the public after these photos have been uploaded to various photo hosting services on the web. They also show that unique information like camera serial number can be used to link supposedly anonymous images with their author.

The reviewers all agreed that this is a very large dataset collected to combat an important and interesting problem of unintended privacy leakage online. In addition to the volume of raw data, the results are impressive as well --- many photos include extensive metadata available for anyone to view.

The reviewers agree that while the paper is suitably broad in covering several different aspects of the privacy concerns regarding photo metadata, this line of work could be improved by performing a deeper analysis of the most potentially harmful privacy vulnerabilities, particularly reidentification.

Privacy is Not an Option: Attacking the IPv6 Privacy Extension
Johanna Ullrich; Edgar Weippl

This paper analyzes the IPv6 privacy extensions for rotating addresses used by clients to make connections, as described in the RFC 4941. The authors find that the specification's randomization, which is used on millions of Windows machines, is flawed and easily broken (though the alternatives used by OSX and Linux are safe). They present a new implementation that follows the specification as closely as possible while increasing security.

Although this IPv6 extension has received a good deal of scrutiny, this paper managed to identify a flaw. As the number of machines using IPv6 is only increasing, it is critical to find these flaws as soon as possible. In addition, IP based tracking is so prevalent in the IPv4 Internet that discovering an analog for IPv6 is timely. A limitation of the attack is that it does take several years of observations; however the attack will become

quicker and cheaper over time, and the paper drives home that either the specification or implementations of it need to change.

Session 8: Evaluating Solutions

Evaluation of Intrusion Detection Systems in Virtualized Environments Using Attack Injection
Aleksandar Milenkoski; Bryan D. Payne; Nuno Antunes; Marco Vieira; Samuel Kounev; Alberto Avritzer; Matthias Luft

hInjector is a framework which aims to expose weaknesses in hypervisor hypercall interfaces. hInjector crafts hypercall based attacks and injects these attacks to test the effectiveness of existing intrusion detection systems which monitor the hypercall interfaces.The paper discusses the authors' criteria for different attack modus operandi, and the evaluation presents case studies which highlight the hInjector's ability to exercise the detection performance of an existing IDS.

Overall, the PC found hInjector to be well-designed and useful in practice for evaluating the detection capabilities of existing IDSes in this space as well as future research into hypercall IDS. This consensus was strengthened by the availability of the author's source code, which will undoubtedly drive future research in this direction. Further, the PC appreciated the authors' well-written explanation of the problem and their methodology for the design and testing of hInjector. Lastly, the PC members would like to see further evaluation of hInjector, potentially with various IDSes, to highlight new observations or solutions that hInjector may be applicable to.

In summary, this is a well-written paper and was well received as it addresses an open problem in the field of virtual environment hardening. The authors do a good job describing the problem and hInjector's design, and the release of their source code is much appreciated. It is expected that hInjector will be useful for both testing existing IDSes and for researchers to evaluate new hypercall IDSes.

Security Analysis of PHP Bytecode Protection Mechanisms
Dario Weißer; Johannes Dahse; Thorsten Holz

This paper analyzes the (in)security of three commercial PHP obfuscators. It proposes static and dynamic techniques to bypass the obfuscators and recover source code by decompiling the protected bytecode. The described techniques are evaluated on 10 obfuscated applications where the authors show that the techniques can be used to recover the original source code.

The PC enjoyed reading about the evaluation of the PHP obfuscators and appreciated the work that went into developing a decompiler capable of deobfuscating the code. It would be great if the decompiler could be made available for academic research so that other researchers can build upon the significant engineering effort that went into producing the described decompiler. As part of future work, PC felt it would be good to include a more formal description of the techniques used to discover code patterns such as jumps and loop structures.

Radmin: Early Detection of Application-Level Resource Exhaustion and Starvation Attacks
Mohamed Elsabagh; Daniel Barbara; Daniel Fleck; Angelos Stavrou

Radmin is a system to detect resource exhaustion or starvation attacks at runtime by building probabilistic finite automata based on resource utilization during benign runs. After an offline training phase to gather such legitimate profiles, Radmin enters an online phase in which instrumented processes are monitored to see whether or not they behave similar to the legitimate profiles.

The PC greatly appreciated the research goal, as resource exhaustion attacks are a severe threat. This well-written paper not only describes a technique to detect such attacks, but also a prototype implementation on Linux. In future work, it would be good

to justify better the accuracy and completeness of the model of legitimate behaviour. In particular, it would be nice to see if repeated measurements of the metrics (such as kernel time, user time and requested resource value) are sufficient for describing the behaviour of the process and for deciding whether it is currently under attack. The results in the paper are clearly encouraging and could serve as a starting point for a more in-depth exploration.

Finally, while a more extensive evaluation (and/or an open source release of the prototype) would have been great --- both to prove the practicality of the approach and for reproducibility --- the idea itself was considered exciting and the PC felt that the paper makes for a nice contribution to the program.

Towards Automatic Inference of Kernel Object Semantics from Binary Code
Junyuan Zeng; Zhiqiang Lin

Argos is a system that recovers the semantics of kernel objects by observing how these objects are used. From annotations on kernel system calls and APIs, as well as knowledge of kernel memory allocation and deallocation routines, bit vectors are built that characterize the sets of operations performed on kernel objects on a per-system call basis. Test cases are used to dynamically exercise the kernel under analysis, and the resulting bit vectors are then compared against hand-crafted filters to identify and characterize kernel objects.

The reviewers universally agreed that the paper was well-written and structured, and tackles an interesting problem. In particular, the reviewers agreed that the paper could have several strong practical benefits, e.g., by enabling more effective analysis of previously unseen operating system kernels, or highlighting critical objects that should be monitored for integrity by kernel defenses.

Several reviewers wondered whether the analysis could be scoped to a subset of the system call API, and whether further kernel API knowledge could be used to improve the semantic labeling of objects. Future work might also consider investigating the portability of these techniques to more platforms, such as Windows. Finally, the reviewers would also have liked more detail on the runtime resource overhead of the analysis, whether that could be improved, and how.

Session 9: Attack Detection II

BotWatcher: Transparent and Generic Botnet Tracking
Thomas Barabosch; Adrian Dombeck; Khaled Yakdan; Elmar Gerhards-Padilla

This paper proposes Botwatcher, a system for the analysis of botnet behavior. The approach leverages VM introspection techniques to monitor low level events triggered by the operation of a botnet on an infected host. A set of inference rules is proposed to correlate extracted execution events into high level activities. The system is evaluated over a number of case studies.

The paper is an interesting read, and while the core problem has been addressed in past work the PC particularly appreciated the blackbox approach idea presented here. The approach is technically simple but elegantly avoids making any assumption on the inner workings of the different botnets. The inference rules are a step forward in the challenging problem of "explaining" low level behaviors into high level activities. The case studies are very interesting and make a nice addition to the paper.

Extending the analysis period to months or years to gather insights on the botnet dynamics is an immediate next step for this line of research. However, the black box approach causes problems when dealing with the parallel operation of multiple droppees, as it is likely to happen when running a botnet for longer periods of time. This is an interesting challenge that appears to be a natural follow up to what is being

presented in this work.

Elite: Automatic Orchestration of Elastic Detection Services to Secure Cloud Hosting
Yangyi Chen; Vincent Bindschaedler; Xiaofeng Wang; Stefan Berger; Dimitrios Pendarakis

Elite is a cloud security framework that allows cloud users to request and deploy anomaly detection services in an elastic cloud infrastructure. Elite involves modifications to OpenStack and employs Spark instances to perform streaming processing of VM logs for anomaly detection. Being elastic itself, Elite is able to dynamically provision more detection instances on demand. Via behavior training and profiling, Elite supports the reuse of previous training results for more efficient instantiation of detection services across users.

The PC found this paper well-written and well-executed, with a solid system implementation. In particular, the idea of elastic, reusable anomaly detection service aligns well with the on-demand, scaling-out properties of today's cloud infrastructures. The application of state-of-the-art cloud technologies, such as OpenStack and Spark, enhances Elite's potential for real-world adoption and deployment. The PC looks forward to the authors' future efforts in introducing next-generation, cloud-wide anomaly detection capabilities that go beyond the traditional, system call tracing-based techniques. They would also be interested in an in-depth discussion on the potential vulnerability (and remedy) of behavior profile sharing between cloud users. Finally, an evaluation of larger scale, with respect to number of instances, requests, and events, would further strengthen the paper.

In summary, this is a strong paper which addresses a timely problem – elastic cloud security. It reflects substantial efforts in system design and implementation and shows potential for real-world adoption. The paper is expected to motivate new solutions to cloud-based anomaly detection, robust in-cloud behavior profiling, and elastic security service provisioning.

AmpPot: Monitoring and Defending Against Amplification DDoS Attacks
Lukas Krämer; Johannes Krupp; Daisuke Makita; Tomomi Nishizoe; Takashi Koide; Katsunari Yoshioka; Christian Rossow

Reflection attacks have become one of the most popular sources for massive scale DDoS. Recent papers have demonstrated that such attacks using a range of protocols are possible; however, little information is known about how widespread such attacks are (at least, from the protocol-specific perspective). This paper is a large-scale measurement study, conducted over five months in 2015 using a series of distributed honeypots.

The authors build and evaluate a honeypot catered towards monitoring and defending against amplification DDoS attacks. AmpPot is built to identify all likely amplification attacks and identified some common features of said attacks, namely: they stem from single sources, and are short-lived. After describing ways to identify publicly available systems that can help towards the amplification attack, the authors describe a honeypot system that can be used to monitor a block of IP addresses, for both proactive and reactive purposes, regarding DDoS attacks. The data collected using the honeypot, allows further investigation of the motives and the attack cases, raising questions for the goal of the attacks. Lastly, the paper discusses how Linux based bots are used to perform DDoS attacks via the amplification technique.

The reviewers appreciated the fact that the paper shed light around the increasingly popular amplification attacks. Having a reliable method for studying them is an important contribution for the community. The authors took care to cover the likely possibilities with respect to amplification attacks. Overall, the reviewers applauded the authors for the care taken to ensure the delivery of their research goals in a responsible way. Making the data available to the community was perceived very favorably, as it increases the transparency and repeatability of this research effort.