SAIS 2019 Proceedings

Southern (SAIS)

3-22-2019

Towards a Virtual Machine Introspection Based Multi-Service, Multi-Architecture, High-Interaction Honeypot for IOT Devices

Cory Nance
Dakota State University, cory.nance@trojans.dsu.edu

TOWARDS A VIRTUAL MACHINE INTROSPECTION BASED MULTI-SERVICE, MULTI-ARCHITECTURE, HIGH-INTERACTION HONEYPOT FOR IOT DEVICES

Cory Nance
Dakota State University
cory.nance@trojans.dsu.edu

ABSTRACT

Internet of Things (IoT) devices are quickly growing in adoption. The use case for IoT devices runs the gamut from household applications (such as toasters, lighting, and thermostats) to medical, battlefield, or Industrial Control System (ICS) applications that are used in life or death situations. A disturbing trend for IoT devices is that they are not developed with security in mind. This lack of security has led to the creation of massive botnets that are used for nefarious acts. To address these issues, it's important to have a good understanding of the threat landscape that IoT devices face. A commonly used security control to monitor and gain insight into threats is a honeypot. This research explores the creation of a VMI-based high-interaction honeypot for IoT devices that is capable of monitoring multiple services simultaneously.

Keywords

IoT, honeypot, Virtual Machine Introspection, VMI

INTRODUCTION

IoT (Internet of Things) devices are becoming more and more prevalent in everyday life. It is estimated that there will be almost 31 billion IoT devices online by 2020 (Statista.com, 2018). The use cases for IoT devices are very broad and consist of anything from critical healthcare devices to TV digital video recorders (DVR) (Abera et al., 2016; Habibi, Midi, Mudgerikar, & Bertino, 2017). These devices differ from a traditional network of computers based on how a user interacts with them. Depending on the use case of the IoT device, once it is set up there is very little interaction from the user (Williams, McMahon, Samtani, Patton, & Chen, 2017). Unfortunately, security has not been a top priority for IoT device manufacturers during specification and design. A recent study from HP assessed vulnerabilities in the most popular IoT devices and found on average each device contained 25 vulnerabilities (Abera et al., 2016). To better learn about the threat landscape facing IoT devices, a honeypot system can be deployed.

An ideal IoT honeypot would be high-interaction and allow for the simultaneous inspection of many services running on the device in order to easily detect when an attack is occurring. However, current research shows that a general high-interaction honeypot is difficult to develop for IoT devices because of the heterogeneous nature of the architectures that the devices are built on (Luo, Xu, Jin, Jia, & Ouyang, 2017). Furthermore, research that focuses on multi-architecture honeypots only target a single service (e.g. telnet, ssh) (Pa et al., 2016; Sentanoe, Taubmann, & Reiser, 2017). In response to this problem, this study proposes to investigate how a high-interaction honeypot can be developed to leverage Virtual Machine Introspection (VMI) in order to monitor multiple services running on the device. This research seeks to make the following contributions:

1. Define a framework for a High-Interaction IoT Honeypot that is capable of monitoring multiple services at the same time.
2. Create an implementation of the framework that can be studied.
3. Conduct case studies by deploying the framework implementation in real world scenarios.

IOT THREATS

IoT threats are created by careless program design (Z. K. Zhang et al., 2014) and not following traditional security recommendations, such as changing the default password on a device. The Open Web Application Security Project (Miessler & Smith, 2018) compiled the top 10 vulnerabilities for 2018 in their Internet of Things Project as shown in table 1 below. Most of the vulnerabilities are not highly technical in nature and could be mitigated by including security in the product development lifecycle. It is speculated that the lack of security in IoT devices is due to manufactors unwillingness to spend the money necessary to secure them (Udemans, 2018).

OWASP Top 10 IoT Vulneratibilites for 2018	
1. Weak, Guessable, or Hardcoded Passwords	6. Insufficient Priacy Protection
2. Insecure Network Services	7. Insecure Data Transfer and Storage
3. Insecure Ecosystem Interaces	8. Lack of Device Management
4. Lack of Secure Update Mechanism	9. Insecure Default Settings
5. Use of Insecure or Outdated Components	10. Lack of Physical Hardening

Table 1. OWASP Top 10 IoT Vulnerabilities (Miessler & Smith, 2018)

One of the most prevalent types of threats to IoT devices are botnets (Habibi et al., 2017). Botnets are defined as “[a] collection of compromised hosts that are under the remote control of a master (aka botmaster)” (Haddadi & Zincir-Heywood, 2015). They run malicious software that connects back to a command and control server and waits for instructions from the botmaster. The botnets themselves can have many different motivations, such as “email spam delivery, distributed denial-of-service (DDoS) attacks, password cracking, key logging, and crypto currency mining (Bertino & Islam, 2017).” IoT devices are prime candidates for becoming bots in a botnet due to their intrinsic nature of being internet connected and vulnerable to a number of exploits.

A recent example of an IoT focused botnet is the Mirai botnet. In late 2016 and 2017, it swept through the internet causing major disruptions to various parts of the internet infrastructure. Notably, on October 21, 2016 the Mirai botnet issued a DDoS attack against Dyn Corporation that resulted in many popular internet sites (e.g. Twitter, Netflix, Spotify) effectively becoming unreachable (Gardner, Beard, & Medhi, 2017). This particular attack generated 1.2 Tbps of traffic and was the largest recorded DDoS attack at the time it occurred (Gardner et al., 2017).

Mirai worked by employing it’s bots to scan the internet for other vulnerable bots (Margolis, Oh, Jadhav, Jeong, & Ho Kim Jeong Neyo Kim, 2017). The vulnerability that Mirai took advantage of was simple; each bot would try to brute-force login via telnet or SSH to a potentially vulnerable device (Kolias, Kambourakis, Stavrou, & Voas, 2017). If it was successful in gaining access, then it would transmit the credentials back to a separate server that was used to infect the victim with the Mirai malware. Once infected, each bot would connect to a C2 server and wait for commands while also scanning the internet for new victims.

Another botnet that plagued the internet in 2017 was BrickerBot (Radware, 2017). As its name implies, it literally “bricks" the IoT devices that it infects. In this case “bricking” means that the device is rendered functionally equivalent to a brick and is not fixable under normal means (Hoffman, 2016). The motivation behind BrickerBot was to act as a form of “Internet Chemotherapy” to force industry to focus more on security (Radware, 2017). The attack vector used by BrickerBot was similar to Mirai’s – it logged in via SSH using a list of common usernames and passwords (Kolias et al., 2017). Once inside, it would proceed to write random data over all the storage disks; effectively turning the IoT device into a brick. As a consequence, unsuspecting customers had their IoT devices incapacitated.

Later in 2017, the Reaper botnet extended Mirai by exploiting software vulnerabilities rather than guessing default credentials (Greenberg, 2017). The Reaper botnet affects many different IoT devices, ranging from routers to IP cameras (Greenberg, 2017). Strangely, this botnet hasn’t displayed any type of DDoS activity yet. However, Mirai showed us that once an attack is set into motion the effects can be devastating. It is estimated that over 1 million organizations are already affected by the Reaper botnet and that we are currently experiencing the calm before the storm (Check Point Research, 2017).

Given the amount of attacks and ease of exploiting vulnerable IoT devices there is a strong likelihood that these types of attacks will increase. In Cisco’s 2018 Annual Cybersecurity Report, it was predicted that IoT attacks will continue to increase (Cisco, 2018). The security community needs to be able to stay on top of new IoT malware and understand how it propagates. A honeypot is a proven way to harvest information on how attackers gain access to devices as well as what malware is being deployed after access is gained (Baumann, 2002; Fraunholz, Krohmer, Anton, & Dieter Schotten, 2017).

HONEYPOTS

A honeypot is an information systems resource that is set up for the express purpose of being attacked (Spitzner, 2003). Unlike most IT assets, honeypots are intended to be attacked and compromised (Baumann, 2002). The value gained from a honeypot depends on the use case. Generally speaking, a honeypot is used to provide intelligence. That intelligence could be alerting an organization to a potential attacker in their production network or gathering the latest malware and attack techniques being used by the black hat community.

At a high-level, honeypots are classified by the environment they operate in – either production or research. A production honeypot is ran inside a production network and is used to alert on attackers that have breached the security perimeter (Verma, 2003). These honeypots do not provide as much attack details as their research counterparts because their primary objective is to perform risk mitigation (Loreto, 2014; Mokube & Adams, 2007). Their value lies in the ability to alert security team members of a possible attack inside their network. Therefore, they are typically only deployed inside companies or organziations. A research honeypot on the otherhand, is typically ran by Universities, governments, militaries, or security companies who want to gain intelligence on attacker methods and techniques (Loreto, 2014; Mokube & Adams, 2007). The data provided by research honeypots can be used to better understand how attackers operate and the exploits being used.

Another way to classify honeypots is by the level of interaction they have with an attacker. Interaction levels are classified as either low, medium, or high. Low-interaction honeypots only emulate part of a service, such as the network stack (Provos, 2003). They do not actually run any commands on a real Operating System (OS), which is an advantage for the security of the honeypot. However, this comes with the disadvantage that it isn't possible to see how an attacker would interact with the OS (Baumann, 2002). A medium-interaction honeypot provides application layer virtualization (Wicherski, 2006). These honeypots do not fully implement all the details of an application protocol, instead they implement just enough to be able to trick an attacker into sending their payload (Wicherski, 2006). Similar to low-interaction honeypots, there isn't an ability for an attacker to interface with an actual OS; however, they have more to offer than a low-interaction honeypot (Loreto, 2014). A high-interaction honeypot gives more freedom to the attacker by allowing access to a real service or even entire OS (Baumann, 2002). This allows attacks to be observed in a realistic setting (Guarnizo et al., 2017). Since none of the services are emulated, high-interaction honeypots have the advantage of being able to uncover new exploits and vulnerabilities (Loreto, 2014). All three types can give researchers valuable information related to how attacks are being executed, what commands or exploits are used, and what malicious software is being ran.

IOT HONEYPOTS

IoTPOT is a high-interaction honeypot that was designed to mimic IoT devices and analyze malicious binaries (Pa et al., 2016). At the time, IoT was seen as an important new area of security research and there was a need to investigate IoT device compromises. IoTPOT consisted of two high level components; a low-interaction front-end responder and a high-interaction backend virtual environment. The front-end was named IoTPOT and acted as a cache; storing the response from the high-interaction backend when a new command was seen. The backend was named IoTBOX and is capable of supporting 8 different CPU architectures due to its use of QEMU; a full system emulator, capable of supporting computer architectures that are normally found in IoT devices, such as ARM and MIPS. Since the backend was high-interaction, outgoing connection attempts were rate-limited to mitigate potential DDoS attack activity.

IoTPOT was allowed to run for 81 days and saw 481,521 malicious download attempts from 79,935 visiting IP addresses. Of the 481,521 malicious files, there were 106 unique malicious binaries. 88 of the 106 had never been seen by VirusTotal[1] before. In the analysis of results, it was found that there were 5 distinct malware families that are being spread by telnet, and most were used to perform DDoS attacks and further spread the malware. An important aspect of the results was that most existing honeypots were low-interaction and would not had been able to capture the binaries that IoTPOT did. For example, honeyd (Provos, 2008) is unable to correctly respond to `echo` request or request to `cat` certain files. This is a major limitation of low-interaction honeypots. Since IoTPOT used a high-interaction back-end, it was able to process the commands correctly and return the results back to the attacker. Another novel aspect of this research was that QEMU was used for the high-interaction backend, allowing for multiple computer architectures to be used as the high-interaction honeypot.

Another IoT focused honeypot is IoTCandyJar (Luo et al., 2017). This honeypot is unconventional because it isn't a traditional high-interaction or low-interaction honeypot. Instead, it uses machine learning to decide what the best response is for a given request. By crowd sourcing IoT devices on the internet with pieces of already seen conversations from their honeypot, it's possible to use a machine learning model that can accurately predict an appropriate response. This technology was dubbed *Intelligent-Interaction*. The system consists of a couple high-level pieces. First is the IoTScanner, which would scan the internet and probe IoT devices with requests that were captured by their honeypot. This new knowledge gets stored and later mined by another component known as the IoTLearner. The IoTLearner is tasked with taking a request and matching it with an appropriate response. This could be challenging because there can be many valid responses but only a few of them are the correct response that will encourage the attacker to continue their attack. During testing, it was shown that the IoTScanner was effective at harvesting replies and identified numerous preliminary checks that are used by attackers. For example, HTTP responses were found to leak data as well as IoT-specific protocols such as HNAP (Home Network Administration Protocol).

[1] https://www.virustotal.com/

Additionally, a common tactic employed by IoT botnet malware is to `echo` a random string. If the response from the honeypot is not the same random string, then the attacker knows that they are connected to a honeypot. To get around this, logic was added to make sure that the argument sent with `echo` is used in the response.

The lack of emulators for IoT devices makes it difficult to create high-interaction honeypots (Luo et al., 2017). Furthermore, it is very time consuming to create low-interaction honeypots for IoT devices due to their heterogeneous nature (Luo et al., 2017). A possible solution is to use QEMU, to emulate an IoT device (Pa et al., 2016). Additionally, Virtual Machine Introspection (VMI) can be leveraged to provide full system functionality and artifacts related to malicious activity (Sentanoe et al., 2017). VMI has demonstrated success in many different security applications ranging from Intrusion Detection Systems (IDS) (Garfinkel & Rosenblum, 2003) to malware binary analysis systems (Taubmann & Kolosnjaji, 2017).

VIRTUAL MACHINE INTROSPECTION

Virtual Machine Introspection (VMI) allows a hypervisor or Virtual Machine Manager (VMM) to inspect the state of its guest virtual machines (Garfinkel & Rosenblum, 2003). It has been shown effective in many different applications, such as intrusion detection systems (IDS), honeypots, and dynamic binary analysis frameworks (Garfinkel & Rosenblum, 2003; Henderson et al., 2014; Sentanoe et al., 2017; Taubmann & Kolosnjaji, 2017). VMI is unique due to its ability to overcome visibility, reliability, and isolation issues that plague a traditional IDS system, because it does not require any modifications or guest agents be installed on virtual machines that are being inspected (X. Zhang, Li, Qing, & Zhang, 2008).

VMI allows for the ability to get access to low-level data such as instructions being executed by the vCPU or the contents of RAM, but it isn't capable of assigning meaning to any of that low-level data that it collects. An open research question surrounding VMI is how to solve the semantic gap problem? The semantic gap can be defined as the process of turning low-level information into high-level semantic information (Dolan-Gavitt, Leek, Zhivich, Giffin, & Lee, 2011). There have been many systems developed to try and solve the semantic gap (Dolan-Gavitt, Leek, et al., 2011; Fu & Lin, 2012; Hizver & Chiueh, n.d.). The two approaches used most often are system call analysis and memory analysis.

System call analysis is done by hooking system calls that are being processed by the Virtual Machine Monitor (VMM). The first system to use this technique was Livewire (Garfinkel & Rosenblum, 2003). Livewire implemented an Intrusion Detection System (IDS) that used VMI rather than a traditional network-based intrusion detection system (NIDS) or host-based intrusion detection system (HIDS) approach. In a NIDS system, the IDS has a complete view of the network traffic and is highly resistant to attack but it does not offer any visibility into what is happening on host systems. In contrast, a HIDS system has a complete view of the host but is not resistant to attacks due to having to install an agent on the host being monitored. Using a VMM, Livewire was able to leverage VMI to give itself a high attack resistance and the ability to see what is happening on the host without installing an agent on the system.

The other common approach used to bridge the semantic gap is memory analysis. The semantic gap problem facing VMI is directly comparable to the semantic gap problem faced by those in the field of forensic memory analysis; therefore, the same solutions being used in digital forensics can also be directly applied to VMI systems. The only difference is that with VMI the memory being analyzed is dynamic rather than static because it is being used by the active VM. In 2011, the semantic gap problem was examined from the viewpoint of digital forensics (Dolan-Gavitt, Payne, & Lee, 2011). A FUSE filesystem was implemented that provided access to guest VM memory. Additionally, a Python C library was created that provided low level programmatic access to the guest VM's memory. This allowed for easy prototyping and integration with existing forensics tools such as Volatility, that are written in Python. Lastly, an extension for Volatility was developed to take advantage of VMI's access to the CR3 register in order to prevent the typical process of having to fully scan memory before being able to extract the initial page table.

VMI HONEYPOTS

Currently, there are only a few VMI-based honeypots in the literature. In 2012, Lengyel et al introduced VMI-Honeymon in an effort to revisit hybrid honeypots since the advent of practical VMI (Lengyel, Neumann, Maresca, Payne, & Kiayias, 2012a). To bridge the semantic gap, VMI-Honeymon analyzed the virtual machine's memory using a common forensics tool; similar to the approach taken a year earlier in (Dolan-Gavitt, Leek, et al., 2011). Their design used the Xen hypervisor with LibVMI and Volatility. It also leveraged a previous work called honeybrid (Berthier, 2015) to create inspection modules that would allow the different pieces of their architecture to communicate. The honeybrid modules were used to ensure that only one high-interaction honeypot was used at a time and that previously seen IP addressed would be filtered out. Additionally, a sandbox execution timer was set on the honeypot's runtime to prevent runaway execution. Over a 2-week period, VMI-Honeymon was able to capture and analyze 2,297 malware samples. Of those, 71% were unclassified by antivirus vendors at the time. It was found that VMI-Honeymon captured 25% more samples than a low-interaction honeypot.

A similar approach to VMIHoneymon was taken when a VMI-based SSH honeypot (Sentanoe et al., 2017) was created. Their design focused on the SSH service and an architecture that consisted of 3 virtual machines (VMs) – a sandbox VM, an introspection VM, and a database VM. The database VM was used to store execution traces from the sandbox VM that were gathered by the introspection VM. LibVMI was leveraged to get running processes from memory and insert software breakpoints for system calls. Once the breakpoints were hit, the parameters being passed to the system call were extracted and recorded. With this technique, VMI was able to extract all the relevant artifacts from memory to reconstruct a decent view of the activity taking place on the machine. The approach used with this VMI-based SSH honeypot was also more effective than existing SSH honeypots due to its ability to detect backdoor connections. VMI enabled the honeypot to get a full, clear picture of the system; including any backdoor-type connections that were established.

FUTURE WORK

This research is currently in-progress. A literature review has been conducted and a gap has been identified in IoT honeypots that use VMI. Based on that gap, this research proposes to create a general-purpose high-interaction VMI-based honeypot for IoT devices. Once completed, this study will benefit security researchers by enabling them to better understand the IoT threat landscape. Companies will also be able to apply this research to better understand threats directed towards their own networks or to provide alerting in a production environment.

REFERENCES

1. Abera, T., Asokan, N., Davi, L., Koushanfar, F., Paverd, A., Sadeghi, A.-R., & Tsudik, G. (2016) Invited - Things, trouble, trust, *In Proceedings of the 53rd Annual Design Automation Conference on - DAC* '16, 1–6. https://doi.org/10.1145/2897937.2905020
2. Baumann, R. (2002) White Paper: *Honeypots*.
3. Berthier, R. (2015) Advanced Honeypot Architecture for Network Threats Quantification, University of Maryland College Park. Retrieved from https://drum.lib.umd.edu/bitstream/handle/1903/9204/Berthier_umd_0117E_10310.pdf?sequence=1
4. Bertino, E., & Islam, N. (2017) Botnets and Internet of Things Security, *Computer*, 50, 2, 76–79. https://doi.org/10.1109/MC.2017.62
5. Check Point Research. (2017) A New IoT Botnet Storm is Coming, *Check Point Research*. Retrieved July 9, 2018, from https://blog.checkpoint.com/2017/10/19/new-iot-botnet-storm-coming/
6. Cisco. (2018) Cisco 2017 Annual Cybersecurity Report.
7. Dolan-Gavitt, B., Leek, T., Zhivich, M., Giffin, J., & Lee, W. (2011) Virtuoso: Narrowing the Semantic Gap in Virtual Machine Introspection, *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/SP.2011.11
8. Dolan-Gavitt, B., Payne, B., & Lee, W. (2011) Leveraging Forensic Tools for Virtual Machine Introspection, 1–6. https://doi.org/http://hdl.handle.net/1853/38424
9. Fraunholz, D., Krohmer, D., Anton, S. D., & Dieter Schotten, H. (2017) Investigation of cyber crime conducted by abusing weak or default passwords with a medium interaction honeypot, 2*017 International Conference on Cyber Security And Protection Of Digital Services, Cyber Security 2017*. https://doi.org/10.1109/CyberSecPODS.2017.8074855
10. Fu, Y., & Lin, Z. (2012) Space Traveling across VM: Automatically Bridging the Semantic Gap in Virtual Machine Introspection via Online Kernel Data Redirection. https://doi.org/10.1109/SP.2012.40
11. Gardner, M. T., Beard, C., & Medhi, D. (2017) Using SEIRS Epidemic Models for IoT Botnets Attacks, *Preceedings of DRCN 2017-Design of Republic Communication Networks, 13th International Conference*, 62–69. Retrieved from http://sce2.umkc.edu/csee/dmedhi/papers/gbm-drcn2017.pdf
12. Garfinkel, T., & Rosenblum, M. (2003) A Virtual Machine Introspection Based Architecture for Intrusion Detection. *NDSS'03*, 1, 253–285. https://doi.org/10.1109/SP.2011.11
13. Greenberg, A. (2017) The Reaper Botnet Could Be Worse Than the Internet-Shaking Mirai Ever Was. Retrieved July 9, 2018, from https://www.wired.com/story/reaper-iot-botnet-infected-million-networks/
14. Guarnizo, J., Tambe, A., Bhunia, S. S., Ochoa, M., Tippenhauer, N., Shabtai, A., & Elovici, Y. (2017) SIPHON: Towards Scalable High-Interaction Physical Honeypots. https://doi.org/10.1145/3055186.3055192
15. Habibi, J., Midi, D., Mudgerikar, A., & Bertino, E. (2017) Heimdall: Mitigating the Internet of Insecure Things, *IEEE Internet of Things Journal*, 4, 4, 968–978. https://doi.org/10.1109/JIOT.2017.2704093

16. Haddadi, F., & Zincir-Heywood, A. N. (2015) Botnet Detection System Analysis on the Effect of Botnet Evolution and Feature Representation, *In Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference - GECCO Companion '15*, 893–900. https://doi.org/10.1145/2739482.2768435
17. Henderson, A., Prakash, A., Yan, L. K., Hu, X., Wang, X., Zhou, R., & Yin, H. (2014) Make it work, make it right, make it fast: building a platform-neutral whole-system dynamic binary analysis platform. *Proceedings of the 2014 International Symposium on Software Testing and Analysis - ISSTA 2014*, 248–258. https://doi.org/10.1145/2610384.2610407
18. Hizver, J., & Chiueh, T.-C. (n.d.) Real-Time Deep Virtual Machine Introspection and Its Applications. https://doi.org/10.1145/2576195.2576196
19. Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017) DDoS in the IoT: Mirai and other botnets. *Computer*, 50, 7, 80–84. https://doi.org/10.1109/MC.2017.201
20. Loreto, J. (2014) The Effectiveness of Honeypots.
21. Luo, T., Xu, Z., Jin, X., Jia, Y., & Ouyang, X. (2017) IoTCandyJar: Towards an Intelligent-Interaction Honeypot for IoT Devices, *Blackhat*. Retrieved from https://pdfs.semanticscholar.org/e44d/3241bdf2d75fee2efc83e5683e852cadfa41.pdf
22. Margolis, J., Oh, T., Jadhav, S., Jeong, J., & Ho Kim Jeong Neyo Kim, Y. (2017) Analysis and Impact of IoT Malware. https://doi.org/10.1145/3125659.3125710
23. Mokube, I., & Adams, M. (2007) Honeypots: Concepts, Approaches, and Challenges. *Proceedings of the 45th Annual Southeast Regional Conference on - ACM-SE 45*, 321–326. https://doi.org/http://dx.doi.org/10.1145/1233341.1233399
24. Miessler, D., & Smith, C. (2018). OWASP internet of things project. *OWASP Internet of Things Project-OWASP*.
25. Pa, Y. M. P., Suzuki, S., Yoshioka, K., Matsumoto, T., Kasama, T., & Rossow, C. (2016) IoTPOT: A Novel Honeypot for Revealing Current IoT Threats, *Journal of Information Processing*, 24, 3, 522–533. https://doi.org/10.2197/ipsjjip.24.522
26. Provos, N. (2003) Honeypot Background. Retrieved July 7, 2018, from http://www.honeyd.org/background.php
27. Provos, N. (2008) Developments of the Honeyd Virtual Honeypot. Retrieved July 12, 2018, from http://www.honeyd.org/
28. Radware. (2017) ERT Threat Alert BrickerBot: Back With A Vengeance BrickerBot.3 – Back With A Vengeance. Retrieved from https://security.radware.com/ddos-threats-attacks/brickerbot-pdos-back-with-vengeance/
29. Sentanoe, S., Taubmann, B., & Reiser, H. P. (2017) Virtual Machine Introspection Based SSH Honeypot, *SHCIS'17*, Neuchatel, Switzerland. https://doi.org/10.1145/3099012.3099016
30. Spitzner, L. (2003) Honeypots: Catching the insider threat. *In Proceedings of Annual Computer Security Applications Conference ACSAC*, 170–179. https://doi.org/10.1109/CSAC.2003.1254322
31. Statista.com. (2018) IoT: number of connected devices worldwide 2012-2025, *Statista*. Retrieved July 7, 2018, from https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/
32. Taubmann, B., & Kolosnjaji, B. (2017) Architecture for Resource-Aware VMI-based Cloud Malware Analysis, *In Proceedings of the 4th Workshop on Security in Highly Connected IT Systems*, 43–48. https://doi.org/10.1145/3099012.3099015
33. Udemans, C. (2018) China's IoT Manufacturers are Reducing Costs at the Expense of Our Privacy and Security. Retrieved February 21, 2019, from https://technode.com/2018/07/02/iot-security-privacy/
34. Verma, A. (2003) Production Honeypots: An Organization's view, Retrieved from https://www.giac.org/paper/gsec/3585/production-honeypots-organizations-view/105831
35. Wicherski, G. (2006) Medium Interaction Honeypots. Retrieved from https://pdfs.semanticscholar.org/9d46/8fa983b844c76a07b1e3ea63d6f7a9cae294.pdf
36. Williams, R., McMahon, E., Samtani, S., Patton, M., & Chen, H. (2017) Identifying Vulnerabilities of Consumer Internet of Things (IoT) Devices: A Scalable Approach, *2017 IEEE International Conference on Intelligence and Security Informatics: Security and Big Data*, 179–181. https://doi.org/10.1109/ISI.2017.8004904
37. Zhang, X., Li, Q., Qing, S., & Zhang, H. (2008) VNIDA: Building an IDS Architecture Using VMM-based Non-intrusive Approach, *Proceedings - 1st International Workshop on Knowledge Discovery and Data Minin*g, *WKDD*, 594–600. https://doi.org/10.1109/WKDD.2008.135
38. Zhang, Z. K., Cho, M. C. Y., Wang, C. W., Hsu, C. W., Chen, C. K., & Shieh, S. (2014) IoT security: Ongoing Challenges and Research Opportunities. *Proceedings - IEEE 7th International Conference on Service-Oriented Computing and Applications SOCA*, 230–234. https://doi.org/10.1109/SOCA.2014.58