ECEN 5053-003 (S'19)

Assignment – 2

[A review of Papers – Submission Choice – 3]

Review Paper

Project Details	*Description*
Created By:	**Rushi James Macwan**
Date	**17th March 2019**
Project Title	**Hands-on with Security**
Class Name	**Developing Industrial IoT** **ECEN 5053-003 (Spring-2019)**
Credits & Courtesy for this work	**BlackHat Publications**
Time spent after this assignment	**10-12 hours**

Credits and Courtesy for this work goes to the individual research publications provided by BlackHat and their respective authors who have published the underlying work which stands as the most technical and relevant information on Computer Security. I have acknowledged the same before basing all my work.

Also, at every occasion, I have tried to cite the work that I have used from outside sources. These citations have been numbered accordingly and can be accessed by clicking the links. I duly acknowledge the credits and courtesy to all of these resources for making this project possible.

Background:

The age of information technology and connected eco-systems have opened up an entirely new world for technological exploitation. With the advancement in certain areas like automation, manufacturing technologies, cyber-physical systems, internet-of-things, cloud computing, cognitive computing which collectively is called as **Industry 4.0** and most importantly the exponential growth in the **Computer Science industry** which we observe by increased connectivity and reduced social proximity has engendered an entirely new set of events. These recent trends of developing technologies have opened a backdoor for "intruders"/imposters from time to time to steal, exploit or misuse third-party resources for significant or insignificant gains which is essentially a crime. These activities have brought up an entirely new sector – **Computer Security** that is all about protecting the digital space that is virtually dominant in our lives.

Security in the virtual world aka digital space that we use to facilitate most of our activities is at the core of this paper. The paper focusses on some of the recent research publications that are made available to the society by the ***BlackHat Community***. The paper emphasizes on the below given Computer Security topics that deal with the loopholes and discrepancies that are often found in the digital world. **To conclude, the paper will highlight the propinquity that these topics of Computer Security share with the Internet-of-Things and Embedded Systems in general.**

Topics:

1. Remotely attacking system firmware[1]
2. Detecting credential compromise in AWS[2]
3. Software attacks on hardware wallets[3]
4. Detecting malicious cloud account behaviour[4]

Remotely attacking system firmware

Firmware is at the heart of any embedded system. To be abstract, personal computers also employ enough low-level hardware that uses firmware (e.g. CD drives, LCD screen controller, keyboard control, etc.). Often times, firmware drives the entire show on any electronic device/system (typically embedded systems) and is the prime attraction for intruders. This topic deals with the quintessential topic of security breaches associated with a system firmware and essentially bridges the information obtained in the published BlackHat document with that delivered in the class.

How is this topic related with class?

Extensive information was provided during the class on how firmware images are authenticated by using Message Authentication Codes (MACs) and is encrypted while transmission through the insecure channel, this topic focusses on the other significant half of this issue which is often ignored and results in to a system integration fault or in other words a security compromise.

What is Firmware?

The primary question that comes to mind under this topic is about firmware. Firmware is basically the system software that runs on a typical system like a server, desktop computer, industry machines or even an automated car.

What are the security threats associated with Firmware?

While most emphasis today is laid on the security aspects of the high-level system interfaces like internet connection protocols, networking architecture, desktop browser or even a system kernel – the hardware element that we call ‘firmware’ is often left out of the picture. The firmware is typically where the threats begin to appear. Threats to the high-level system interfaces can have consequences of varying security issues like loss of personal information or bank account information. On the other hand, low-level system attacks (i.e. firmware attacks) have a greater threat vector. How this works is defined more clearly in the below sections of this topic.

So, how do low-level firmware attacks take place?

At the core of any firmware, the **Baseboard Management Controller (BMC)** governs, processes and monitors the services on a computer system. It consistently manages the physical and virtual state of the device, networking capabilities and the hardware elements through the effective use of sensor networks (e.g. temperature sensor). In addition, it also widely manages system interfaces (e.g. SPI bus, I2C bus, RAM, MMU, DMA access, etc.). The above processes are realized by proper communication with the system administrator services like BIOS settings, OS provisioning, KVM processes and threads.

So, while attacks span from browsers to kernel in the software domain, the attacks in the firmware domain are related with hardware and low-end server motherboards. Fog computing which primarily employs the idea of a server on a module is the prime hub of such attacks.

Low-end hardware modules contain several parts like CPU, RAM, ROM, Flash memory, PLDs, etc. and the vendor selling these complete hardware modules often receive 10-15 of these components from other vendors who are experts in the said manufacturing. This leads to a situation where the main vendor may not either have an access to the source code (firmware) of the components (often due to IP issues or product version), updates may not be available to the vendor and pushes the product to the consumers.

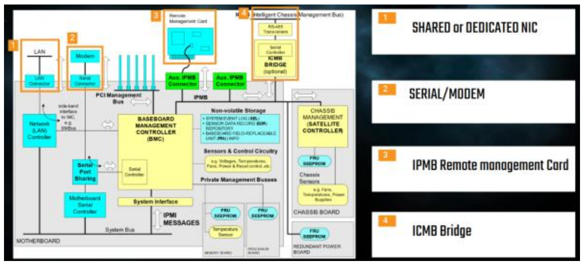
Attacks occur after these devices have fallen into the ends of the consumers who become victim of two attacks:

1. Physical access attacks
2. Software access attacks

Physical access attacks often are related with BMC and/or other aligned technologies like Intel’s ME/AMT and which critically compromises the system hardware. Software access attacks often happen due to open-source attacks which are related with the presence of malware in the firmware that is being updated by the system.

How do BMC or ME/AMT attacks take place?

BMC starts running the moment a typical computer system is powered. It has its own separate CPU, memories, network architecture with its own MAC address and much more. Essentially, it is a small computer that manages the entire system. In addition to all of that, it has its own web-server, DMA capability (which it uses while software updates are in progress), SPI access, etc.



The ME/AMT system like the BMC was released as a chipset by Intel. It typically contains a tiny computer just like the BMC which regulates the processes and flow of any generalized computer system.

To sum up, these hardware based technologies opens up a place for attackers in one of these three areas:

1. KVM (Kernel-based Virtual Machine)
2. OS – installation and/or provisioning
3. BIOS Flash

While the BMC or ME/AMT technology has its own network stack, it performs system provisioning before the execution actually falls into the hands of the system administrator. What that means is that the system is booted and the system status is enabled/disabled. If the status is enabled, it falls into two stages provisioned or un-provisioned. The provisioned mode is where the firmware is locked and is not in the process of updating itself. If the updates are to be carried out, the enabled status (which specifies that the system is running) is un-provisioned and that is where external access to the BMC or ME/AMT system is possible which indirectly leads to an access to the entire computer system – since the BMC or ME/AMT system governs the entire computer. Attackers gain external access through the malware present in the open-source firmware updates and then potentially provision the system – thereby gaining full control over the system since once the system is provisioned, the new firmware is locked into the chipset.

Ultimately, the newly provisioned system is now prone to change in BIOS settings performed by the malware while run-time or boot-time since the BMC or ME/AMT network interfaces communicate with the main processor and with other system components as well.

Courtesy of: [1]

Detecting Credential Compromise in AWS

Cloud (fog) computing services help companies realize the prime goal of adapting to the Industry 4.0 standards. Amazon Web Services (AWS) provide companies/organizations with the ability to connect at a greater level to provide quality services to consumers by connecting elements across an organization. This topic will focus mainly on how security compromises are made in the Cloud Services and how it leads to a potential breach in the efficient functioning of a company.

How is this topic related with class?

During the discussions on Markets, Platforms, Services and Applications in the class, a major emphasis was laid on how companies/organizations connect on a larger scale – to increase operational efficiency, reduce operational expenses, increase sales & profits and most importantly to reduce risks. Organizations thus use platforms that actively manage and control the activities across the organization. Platforms manage devices, applications and networks across the organization. It receives highly customizable APIs that are built and delivered by external suppliers. What this means is that it leaves a gap inside the platforms for intruders to steal the credentials of the authorized third-party suppliers who provide customizable software support and in-turn degrade the value of the Organization. This paper will discuss more on the same in detail keeping the AWS service at the centre of attraction.

What is AWS and how it works?

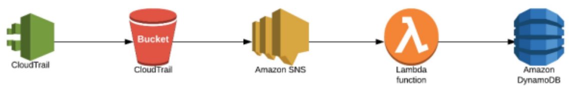
Amazon Web Services (AWS) provides on-demand cloud computing platforms to individuals, companies and potentially to governments on a metered pay-as-you-go basis. AWS connects its services across various companies and organizations that take help from other parties to build and regulate the AWS services for them.[5]

How does the threat appear?

Threats come into picture when intruders are able to either steal the identity of the authorized service providers to any organization or when the intruders openly exploit and modify the existing services while the services are actually compromised.

How does AWS fight with the security issues?

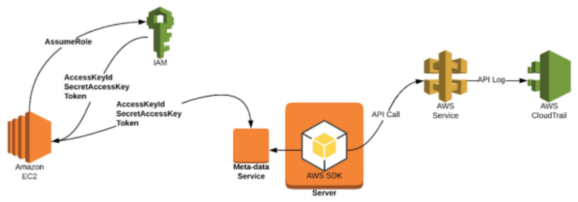
AWS incorporates a CloudTrail Service which keeps a clean record of an organization's AWS account activity, including actions taken through the AWS Management Console, AWS SDKs, command line tools and other services. Basically, it notes the IP addresses of the users of the account (the service-providers to an organization) and thereby keeps a track of parameters that specify their activity (i.e. event source, event name, AWS region, etc.).



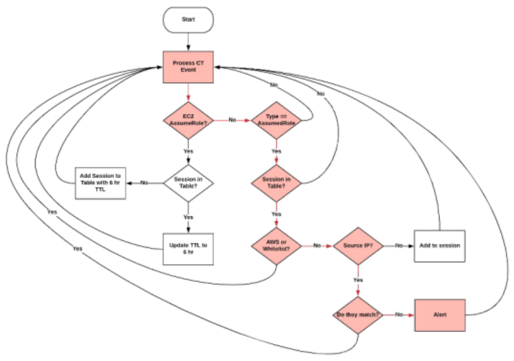
This event-driven feature was implemented as opposed to the polling based feature that "tries" to keep a record of the IP addresses that appear across hundreds of thousands of servers in a given region and fails (one incident wherein an IP address was logged in for 4 minutes and was unreported to the system). Event-driven approach in the CloudTrial Services compare the existing IP addresses with the ones in the database and alerts the AWS account holder if a new IP address is observed. This ensures fair use of the Cloud Computing resources and reduces storage requirements while also providing real-time feedback.

How is the event-driven framework utilized?

AWS uses an Amazon Elastic Compute Cloud (Amazon EC2) that services all the needs of an organization. AWS provides temporary (valid for 6 hours) tokens to service providers for the account of an organization. The access is temporary and has limited-privileged credentials for AWS Identity and Access Management (IAM) for users that the organization authenticates.



This EC2 obtains requests from users to gain temporary access tokens and the EC2 sends key requests to IAM. It provides EC2 with secret keys (as an access token while the IP address is exchanged for acquiring the keys) that the user is given to for accessing the system. In real-time, the CloudTrail keeps a record for the last 90 days of the service period of all the IP addresses that have obtained permission to exploit the services. Thus when EC2 sends the access tokens to the meta-data service module inside the AWS SDK (Server), the server instantaneously forwards the keys (IP addresses that bear those keys) to the AWS Service Platform. The AWS Service Platform matches the obtained keys with the ones present in the AWS CloudTrial Service account of the organization and sends an alert if the match fails. This way, AWS guarantees that security of the Cloud accounts is not compromised. However, Security can never be fully guaranteed and thus there always remains a trade-off since organizations will only have an access to the 90-day service period IP addresses and therefore every IP address that is received after a period of 90-days will generate an alert which will be redundant.



Courtesy of: [2]

Software attacks on hardware wallets

Hardware (HW) wallets are basically crypto wallets that carry tokens, access keys and PINs that provide the authenticated user an access to this information for performing various transactions on cryptocurrency. Today, the growing platform of blockchain and cryptocurrency demands a reliable, secure and efficient way of storing the transactions and private information so that intruders do not gain an access to the cryptocurrency. This topic will focus on the current security issues with the hardware wallets that store the valuable user information pertaining to cryptocurrency transactions.

How is this topic related to class?

A few topics pertaining to computer security were discussed in class that relied on the assumption that physical access of the device was not available to the intruders. While this cannot be always true, the paper will identify some of the ways in which hardware security attacks are made in accordance with hardware wallets as they tend to be highly resource-constrained embedded systems.

What is cryptocurrency and what are hardware wallets and how do they work?

Cryptocurrency is essentially digital money that is protected by a single private key. Hardware wallets are generally secure hardware user devices that the user can trust for storing private information (i.e. private key used for cryptocurrency transactions) that is linked with the user account. Secure HW user devices take one of the following forms:

1. Smart card based banking cards
2. Secure element based solutions
3. General purpose microcontrollers

Smart card based solutions are tamper-proof & cheap, do not require custom hardware, offer multiple applications on HW wallet and most importantly has API for secure crypto implementation.

Secure elements on the other hand differ by the presence of a secure HW crypto engine and allows to implement screens/controls. The downside is that it requires additional hardware to read/implement applications and to control/display certain information.

MCU based HW wallets again have all the benefits from the previous types but essentially have no HW crypto support and are slow. They have simpler hardware design with one MCU with controls/screen and often have fully open HW/SW solution which opens up a backdoor for intruders. Most importantly, MCUs do not have hardware encoding, elliptical curves and hashing of hardware solutions is not possible. Often it uses separate crypto processor which increases price and complicates the design or trade-offs with slower execution.

How does the threat appear in HW wallets?

The threats that appear can be of two types:

1. Logical Attacks
2. Physical Attacks

Logical attacks are often performed by exploiting the system vulnerabilities. This is done by finding the errors in the design/realization/authentication mechanism of a HW wallet. Often, HW wallet manufacturers using MCU solutions will verify the designs without API and vulnerabilities in the authenticated state and PIN verification is left unattended.

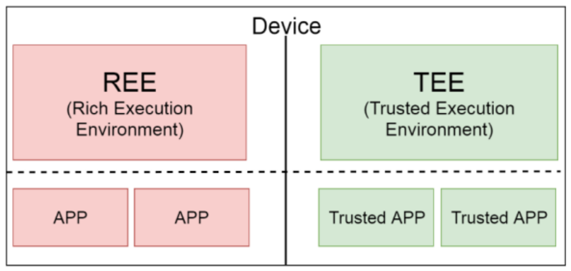
Physical attacks are often carried out by exploiting the electrical characteristics of a device. Simple power analysis can help intruders trace the valuable information that the device is processing at a given moment if the hardware vulnerabilities are left unattended. Fault injection errors which occur when the device is physically tampered produce unpredictable outcomes which may help the intruders to steal private information. An example showed how a 128-bit AES encrypted system was prone to such attacks.

In general, intruders can often break into the security shield of a HW wallet by exploiting one of these two vulnerabilities:

1. Un-authorized operations
2. Bypassing incomplete checks

A quick case-study of a software attack on a hardware wallet:

The Ledger Nano S HW wallet uses a MCU and a secure element ST31. The MCU provides a non-secure service by controlling the USB communication, control buttons, device's screen and communicating with the secure element. The study in the reference document for this topic provides a review of how the secure element uses a private TEE (Trusted Execution Environment) OS while the MCU uses open-source applications. This leads to exposing a software glitch where dereferencing a null-pointer and then feeding it to system call exposes certain firmware information. This basically highlights the underlying vulnerability present in the firmware development.



On the other hand, it was found that while the secure element uses a TEE OS code, most of the code is open source. The OS itself and the wallet applications share the flash memory of the secure element. The OS has higher priority over the wallet applications in the firmware architecture and this essentially leads to a vulnerability. Since the user is essentially required to download and install open-source wallet applications, intruders can provide malicious wallet applications that can overrun other wallet applications (given that OS has higher priority) during run-time & execution. These malicious applications can read the stored information for other wallet applications and can find the location of the key in the shared memory space of the secure element and ultimately leads to a huge vulnerability. What it means is that wallet applications must not be allowed to communicate with each other which is the root cause of data breach.

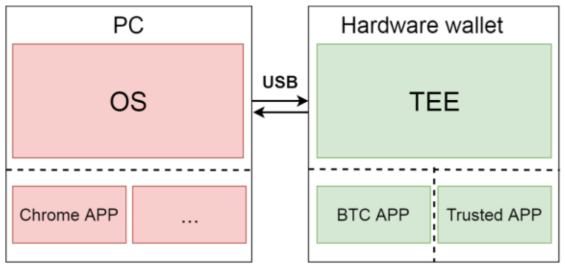
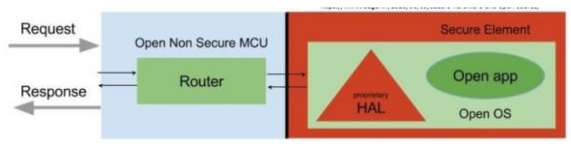


Fig. 1 - TEE design of the hardware wallet



In one more study presented in the same reference, it was found that while the private key to a Bitcoin wallet application remained safe, the vulnerability allowed stealing the secret key from a Monero digital wallet as well as U2F application HMAC key which was originally designed to secure access to online services. Thus, intruders can build malicious wallet applications that can use debug flags in the firmware and potentially read data from misconfigured memory protection units.

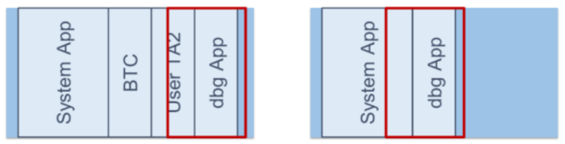


Fig.2 – Memory access of dbgApp to the flash memory

Courtesy of: [3] [6]

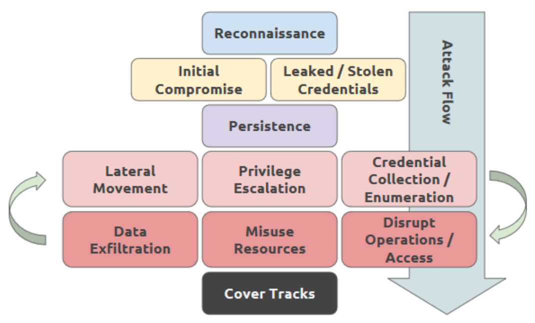
Detecting malicious cloud account behaviour

Malicious cloud account behaviour drives the vulnerability cycle of any potential embedded/IoT system. Malicious activities across the cloud accounts and services provide us a quick understanding of what a system breach might lead to. This topic is more in alignment with the topics presented previously in this paper. This topic will briefly address some of the essential pillars that hold a system potentially safe (although never fully secure).

How is this topic related to class?

This topic relates some essential industry aspects of cloud security with the topics that were discussed in class (e.g. SDNs and NFVs) and the unauthenticated use of these technologies.

Cloud Attack Path/Lifecycle (Adapted)



What typically adds to a cloud vulnerability?

- Access keys are present almost everywhere in the cloud platform.
- Lateral movement – credential collection is highly possible in the cloud platforms.
- Misuse of resources – cryptomining, cryptojacking, etc. adds to the vulnerability.
- Because APIs are same that everyone uses to access the system and the fact that the keys are everywhere, it is possible to disrupt operations/accesses.
- Data extrafiltration adds to the list.

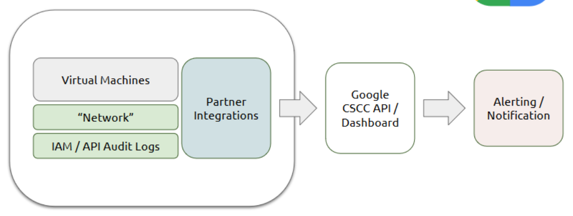
What are the detection resources?

- Network activities – unusual IP addresses, unusual traffic patterns
- DNS (Domain Name Service) issues
- Host-based (OS/Application) issues
- Unusual Cloud Provider API activity
- Web/user access log issues

Case-study on Google Detection Flow:

The Google CSCC (Cloud Security Command Centre) is the canonical security and data risk database for Google Cloud Platform (GCP). Cloud SCC enables you to understand security and data attack surface by providing asset inventory, discovery, search, and management.

Detection Data Flow



Google CSCC provides this service using anomaly detection and identifies the below threats:

1. Botnets - A botnet is a collection of internet-connected devices, which may include PCs, servers, mobile devices and internet of things devices that are infected and controlled by a common type of malware. Users are often unaware of a botnet infecting their system.[7]
2. Cryptocurrency mining – It is a process in which transactions for various forms of cryptocurrency are verified and added to the blockchain digital ledger.[8]
3. Anomalous reboots – This again deals with the basic idea of Botnets which essentially can send malware to dozens of devices for multiple reboots during which the system execution falls into the low-level BMC or ME/AMT hardware (as previously mentioned in the first

topic) and intruders can exploit the device vulnerabilities.

4. Suspicious/anomalous network traffic – This issue is critically linked with Software Defined Networks (SDNs) and Network Function Virtualisations (NFVs). Inconsistent network behaviour could potentially mean a compromise with these services.

Thus, the Google CSCC basically captures a view of all the below elements in a cloud service platform for detecting the anomalies:

1. Virtual Machines
2. Networks
3. IAM (Identity and Access Management) / API Audit logs
4. Partner integrations – Service Providers

Courtesy of: [4]

Conclusion:

What I did?

In this paper, carried out a detailed study of some of the topics that were presented at BlackHat events in the recent past. I pondered upon the implicit relations it has with the growing computer security threats. I tried to relate the topics with the huge picture of Embedded Systems and Internet-of-Things in general.

What I learnt?

As the end of conclusion of this paper, I learnt some of the critical security issues that are present across the Embedded System and IoT devices. To quickly encapsulate, below are some of the key trending areas that I touched upon in this paper:

1. Firmware threats & vulnerabilities
2. Software attacks on typical Embedded Systems
3. Physical & Logical threats to some of the existing Crypto-currency Systems
4. Credential compromises in the Cloud Platforms & Services
5. Critical safety issues and malicious behaviours in the Cloud Platforms & Services
6. A generalized build-up of security mind-set

How in general does it relate to class?

The class sessions have introduced in the past to the existing security issues from the generalized perspective of an Embedded IoT system. This paper elongates the same principles and throws more light on the trending security issues in some of the dimensions that is currently under the microscope (i.e. Firmware, Software, Hardware & Physical issues). More than that, the paper relates the industry implementations of the same root concepts that were discussed in class regarding security – namely AES encryption, asymmetric encryption, hashing, MACs, etc.

Citations

[1] - Remotely attacking system firmware	
[2] - Detecting credential compromise in AWS	
[3] - Software attacks on hardware wallets	
[4] - Detecting malicious cloud account behaviour	
[5] - Amazon Web Services (AWS)	
[6] - Ledger Nano S – Cryptocurrency Hardware Wallet	
[7] - Botnets	
[8] - Cryptocurrency Mining	

Please Note: All images and figures presented in this paper are solely the contents of the authors who have presented the topics at BlackHat events. I do not own them and I duly credit their work in this paper. If otherwise, they belong to the Class Slides and duly credit Prof. David Sluiter (Instructor) for the same. Wherever images and figures are unclear, they can found in the referenced documents.