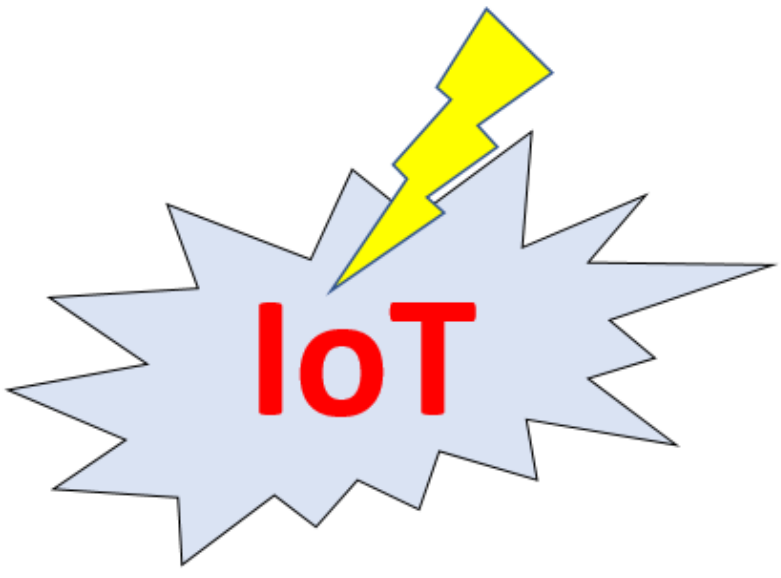
IoT Security - Part 21 (Famous IoT Attacks & Vulnerabilities)

asmita-jha

29-December-2020



This blog is part of the IoT Security series, where we discuss the basic concepts about the IoT/IIoT ecosystem and its security.If you have not gone through the previous blogs in the series, I will urge you to go through those first. In case you are only interested in reading about a few famous IoT attacks and vulnerabilities, feel free to continue.

IoT Security - Part 1 (101 - IoT Introduction And Architecture)

IoT Security - Part 20 (101 - Introduction to Fault Injection Attack (FI))

In our previous parts of this blog series, we discussed the basic overview of IoT security, covering the introduction, attack surface, radio, firmware, IoT protocols, hardware attacks, etc.This is the last part of our journey on IoT Security 101 blog series. We hope this blog series would be a good starting point for someone getting started in IoT security. In this blog, we will discuss a few famous IoT attacks and vulnerabilities to conclude this series. Though there are many IoT attacks with varying attack surfaces ranging from hardware, firmware, radio, etc., we have chosen a few of them for this blog. Kindly note that the sequence in which these attacks have been mentioned here does not convey anything.

Ripple20

JSOF Research labs discovered Ripple20. It’s a series of 19 zero-day vulnerabilities in the low-level TCP/IP library developed by Treck, Inc. This affected library exists in various IoT and embedded devices, including medical, home, enterprise, networking, retail, aviation, power grids, etc., leading various devices at high risks. The risks include data theft, malfunctioning of industrial devices, patching with malicious code/ backdoor in the device, remote control, etc. Hence, the issue in one library created a ripple-effect for the millions of devices in the entire supply chain of various sectors. Out of the 19

vulnerabilities found, 4 are critical remote code execution with CVSS score greater than or equal to 9, 4 are having CVSS score greater than or equal to 7, 11 have various lower severity, DoS, info leaks, etc., and two were anonymously reported. The technical details on a few of its CVEs are mentioned below.
Reference: JSOF website

CVE-2020-11896

- Remote code execution (RCE)
- For performing RCE, the prerequisites for the device is to support IP fragmentation and IP tunneling. If these prerequisites are not met then the vulnerability remains DoS.
- An attacker can perform the RCE on the target device if he/she can send UDP packets to an open port of the device.
- Fixed on version 6.0.1.66
- Reference : JSOF technical details

CVE-2020-11898

- Information leak
- The incoming IPv4 fragments over an IP-in-IP tunnel is not handled properly by the Treck TCP/IP stack.
- An attacker can leak memory from the heap leading to information leakage vulnerability.
- Fixed on version 6.0.1.66
- Reference : JSOF technical details

CVE-2020-11901

- It consists of many critical client-side vulnerabilities in Treck's TCP/IP stack DNS resolver.
- The vulnerability is in the DNS resolver component of the network stack. It occurs from an incorrect DNS label length calculation.
- After the successful exploitation, an attacker can perform RCE to respond to a DNS query generated from an affected device.
- Fixed on version 6.0.1.66
- Reference : JSOF technical details

More details about all the vulnerabilities can be read at JSOF Ripple20 Technical Section

Meltdown and Spectre

Meltdown and Spectre are the hardware vulnerabilities in processors to leak/steal information by abusing CPU data cache timing via side-channel analysis.

Meltdown attack exploits the side effects like leakages via timing differences and cache analysis of out-of-order execution features present in most of the modern processors. More detailed working can be read from the Meltdown research paper mentioned under the reference. It bypasses the memory isolation security feature between the user applications and the operating system that ensures the kernel address

range should be non-accessible and protected from user access. The adversary with non-privileged access can read the entire kernel memory through this attack, including any mapped physical memory without any permission. It was thus leaking critical information of the operating system and the other programs.

Spectre attack exploits the branch prediction and speculative execution feature of processors. These attacks trick the processor to speculatively execute instructions that should not have occurred during the correct program execution. The CPU states are reverted to maintain the functional correctness, which is called transient instructions. The adversary can then leak the sensitive information via side-channel analysis from the processor's memory address space by influencing the transient instructions that are speculatively executed.

There are three known variants of these vulnerabilities, variant 1 : CVE-2017-5753 and variant 2: CVE-2017-5715 is grouped under Spectre attack, and variant 3: CVE-2017-5754is grouped under Meltdown. There are patches to prevent from the known exploits based on Meltdown and Spectre through software patches. References: Meltdown: Reading Kernel Memory from User Space, Spectre Attacks: Exploiting Speculative Execution, Meltdown attack, IoTSF, Ars Technica, The Register, Google Project Zero

Mirai botnet attack on IoT devices

The IoT botnet attack exploits the insecure IoT devices with open Telnet ports and weak/default credentials to build a botnet. IoT devices located at inaccessible locations without the feature to be remotely patched become the victim of such attacks. Mirai) is a malware that turns Linux based network devices into bots to form a large botnet army. It has been used to perform major DDoS attacks including Brian Krebs' website, French web host OVH, Dyn cyberattack. It is used to convert the insecure IoT devices into bots. The infected devices then scan for other vulnerable IoT devices to infect them, thus becoming self-propagating. The creators made the source code for Mirai open source. Since then, it's used to perform attacks on many IoT devices.

References : csoonline, IoTSF, Wikipedia)

Rolljam Attack

This attack, demonstrated by Samy Kamkar, targeted the key fobs of cars that are used to unlock the door remotely and for other purposes. He developed a gadget called "Rolljam," a radio device to perform this attack in the vehicle's radio range. When the door unlock button is pressed on the key fob, it transmits the code via the radio signal that is received by the car. The door gets opened if the code received matches the car's code. The key fobs use a rolling code security mechanism to send these codes where each time a new code is generated and old codes that are already used are discarded to avoid replay attacks by the attacker. But the rolljam attack brokes this security mechanism by recording and jamming the radio signal from the key fob. The car doesn't unlock because the signal is jammed. Hence the owner has to send the signal again. The second signal is also recorded and blocked, but this time the attacker unlocks the door by broadcasting the blocked first code. The adversary now also has the second sequence of recorded code that is not used by the car yet and hence not discarded. The adversary can use it to unlock the door anytime later.

References: Wired article, Research paper titled, "Hold the Door! Fingerprinting Your Car Key to Prevent Keyless Entry Car Theft", Hackster.io article

Sweyntooth vulnerabilities

Sweyntooth is a collection of Bluetooth Low Energy (BLE) vulnerabilities discovered by a group of researchers at SUTD Asset Research group. As of now, it has a family of 18 vulnerabilities. The vulnerabilities are across the various BLE software development kits (SDKs) of major systems on chips (SoCs), exposing the BLE implementation flaws. Depending on the applications, it allows an adversary in the radio range to cause crashes, deadlock; buffer overflows, security bypass, etc. It affected various IoT devices having the vulnerable BLE stack from different vendors.

The CVEs related to these vulnerabilities include : CVE-2019-16336, CVE-2019-17519, CVE-2019-17061, CVE-2019-17060, CVE-2019-17517, CVE-2019-17518, CVE-2019-17520, CVE-2019-19193, CVE-2019-19195, CVE-2019-19192, CVE-2019-19196, CVE-2019-19194, CVE-2020-13593, CVE-2020-13595, CVE-2020-10061, CVE-2020-10069, CVE-2020-13594.

As mentioned in the Sweyntooth github repo, Zero LTK Installation CVE-2019-19194 is the most critical sweyntooth vulnerabilities. It allows the adversary to bypass the latest Bluetooth pairing mechanism altogether. More detail about the other CVEs can be read from the links mentioned in the references. Though many vendors have released the patch as mentioned on the research page,there could still be various unverified devices with these vulnerabilities. To automate these testings, the researchers have released the open-source Sweyntooh Framework with proof of concepts. Reference : SUTD Asset Research Group, SweynTooth: Unleashing Mayhem over Bluetooth Low Energy

Philips Hue smart lightbulb Zigbee vulnerability

The vulnerability was found in the Philip Hue smart bulbs. It exploited the heap-based overflow in the implementation of the Zigbee wireless protocol while handling a long ZCL (Zigbee cluster library) string during the commissioning phase. It resulted in remote code execution in the bridge component that is responsible for sending the remote commands to the bulb over Zigbee protocol. To achieve this, the attacker has to first compromise the bulb with the malicious firmware (this was achieved a few years before by exploiting the vulnerability in the Zigbee Light Link (ZLL) protocol implementation. It can be referred to in the report titled,"IoT Goes Nuclear: Creating a ZigBee Chain Reaction"). The compromisation of the bulb with the malicious firmware makes it appear unreachable from the user control app. It tricks the user into thinking that there is just a fault. To reset the bulb, the user removes it from the app and instructs the bridge to discover the bulb. The bridge discovers the compromised bulb that then gets added to the network. Exploiting the Zigbee implementation vulnerability to trigger the heap-based overflow on the bridge enables the adversary to send malicious code (malware) to the bridge. The infected bridge connects back to the hacker, infiltrates the network using known exploits, and compromises other devices, including the target network's computer network. The vulnerability in a smart bulb leads to the compromisation of the complete target network. The corresponding CVE is : CVE-2020-6007. After the Checkpoint research team's responsible disclosure, the patch has been released. Still, the main issue is in the Zigbee protocol implementation that can attack the other IoT devices in similar ways.

References : Checkpoint research

Fault Injection Attack on ESP32

Researchers at Raelize have discovered various flaws in ESP32 using EM (Electromagnetic) fault injection attacks. The attack, as mentioned in CVE-2019-15894 bypasses the secure boot verification that was later patched by implementing flash encryption. Later, the researchers bypass even this flash encryption and thus the secure boot via fault injection. The corresponding CVE is CVE-2020-15048. These attacks lead to the execution of unverified code, escalating privileges on Linux, arbitrary control of the program counter (PC) register, etc. More detail about these attacks can be read in our previous blogs FI

and SCA. ESP32 is low cost, easy to use. It is used by many hobbyists and also in developing low-cost IoT products. Attacks like these can affect other chips used to develop IoT products that might not be resistant to such attacks.

References : Raelize

BLESA Attack

BLESA, i.e., Bluetooth Low Energy Spoofing Attack, exploits the BLE software stack implementation vulnerability. This vulnerability is related to BLE's reconnection procedure, during which the two previously connected devices reconnect. As per the BLE specification analysis, researchers found that the BLE specification does not mandate it to perform authentication during reconnection. The vendor implementation can take this reconnection authentication part optional despite following the specification hence prone to attack. These critical weaknesses identified in the BLE specification make the BLE devices vulnerable to spoofing attacks. This attack enables the adversary to impersonate the BLE device and provide the spoofed malicious data to the previously paired device. It was found that this attack applied to many BLE stack implementations, including Linux BlueZ, Android, and iOS. Though Apple assigned the CVE-2020-9770 to the vulnerability and fixed it, as per researchers, the Android on which this attack was made is still vulnerable. As many IoT devices have BLE implementation, this is a critical threat if the stack implemented inside them possess the same vulnerability. References : Research Paper titled, "BLESA: Spoofing Attacks against Reconnections in Bluetooth Low Energy", Zdnet Article

Bluetooth attack on Tesla Model X

The recent Bluetooth attack on Tesla Model X's keyless entry system showed that the security researcher named "Lennert Wouters" at Belgian university KU Leuven could break into the car in 90 seconds. The vulnerability in the key fob firmware update mechanism over Bluetooth was exploited, allowing the adversary to patch the key fob with the malicious firmware. It shows the lack of code signing feature in the over-the-air firmware update mechanism of the key fob. The researcher revealed that a group of security vulnerabilities was found in the Tesla Model X car and their key fobs. These vulnerabilities can be combinedly exploited to unlock and steal the car. The malicious firmware in the keyfob was able to query the secure enclave chip responsible for generating the car's unlock code. The details about it can be read from the Wired article mentioned under the references.

References : Wired article

Amnesia:33

Amnesia:33 discovered by the Forescout research team is a collection of 33 vulnerabilities in four open-source TCP/IP stacks (uIP, FNET, picoTCP, and Nut/Net). The vulnerabilities include improper memory management, overflow flaws, lack of input validation, etc. The exploitation of these vulnerabilities could lead to memory corruption, remote code execution, perform DoS (Denial of service0) attacks, information leaks, etc. The TCP/IP stack is used in millions of IoT devices ranging from consumer IoT, industries, power grids, medical, access controls, servers, routers, the whole supply chain, etc. We also saw the Ripple20 vulnerabilities that were found in the TCP/IP stacks. As per researchers, more than 150 vendors and millions of devices are vulnerable to Amnesia:33. Thus these vulnerabilities possess potential threats. There could be possibilities of the presence of vulnerable TCP/IP stacks in many code bases of various products leading to more such attacks.

References : Hackernews article, Forescout research lab, Blackhat

Conclusion

There are a lot more different attacks related to IoT. In this blog, we picked up a few of them to show the various attack possibilities in the IoT ecosystem, ranging from hardware, firmware, radio, etc. We hope you got the idea of the threat on the IoT devices and why it is so essential to gear up IoT security.

About Payatu

Payatu is a research-powered cybersecurity service and training organization specialized in IoT, embedded, mobile, cloud, infrastructure security and advanced security training. We offer a full IoT/IIoT ecosystem security assessment, including hardware, firmware, middleware and application interfaces. If you are looking for security testing services then let's talk, share your requirements: https://payatu.com/#getstarted. Payatu is at the front line of IoT security research, with a great team, and in house tools like expliot.io. In the last 8+ years, Payatu has performed, security assessment of 100+ IoT/IIoT product ecosystems and we understand the IoT ecosystem inside out. Get in touch with us. Click on the get started button below.