Chapter 20

Attack Detection and Countermeasures at Edge Devices

Fahim Ahmed[1] and Md Tanvir Arafin[2]

[1] Cyber Security Engineering Department, George Mason University, Fairfax, VA, USA, email: fahmed32@gmu.edu
[2] Cyber Security Engineering Department, George Mason University, Fairfax, VA, USA, email: marafin@gmu.edu

Abstract: This chapter presents the current security status of edge devices, more specifically, common attacks and countermeasures for devices in the existing Internet of Things (IoT) ecosystems. As edge devices permeate our lives, the potential of catastrophic attacks through edge devices becomes an everyday reality. Recent attacks such as Mirai, BrakTooth, Lemon Duck, and Satori have demonstrated how simple edge devices can be leveraged to launch large-scale attacks. Unfortunately, the countermeasures for these attacks are still immature and not widely adopted. In addition, there is a significant void in vulnerability research for edge devices. Hence, this chapter introduces the state-of-the-art security issues in edge devices and the existing countermeasures from a hardware-centric perspective.

Keywords: Edge Devices, Malware, Botnets, DDoS, Side-Channel Attack, Fault Injection Attack, MUDs, Zero Trust Architecture.

20.1. Introduction

In recent years, the world has experienced a marked explosion in the number of low-power, application-specific computing devices connected via the internet. These devices are mostly connected at the edge of a network for data collection, aggregation, and data-derived computation at a granular level and for providing intelligent system design and control solutions. In current literature, these networked connected components are referred to as *Things* or *edge devices*.

Edge devices solve some of the long-standing problems of intelligent real-time systems design. These devices are ubiquitous in applications – from home appliances to power systems to edge robots. For example, connected sensor nodes (such as phasor measurement units) in a large area power distribution network can provide a detailed picture of the load (*i.e.,* power consumption) within a grid and enable better load-balancing, generation, and control algorithms leading to a *smart* grid. Similarly, edge robots (*i.e.,* small-scale robots and autonomous systems) can be deployed for intelligent rescue operations, autonomous surveillance, and delivery services. Additionally, large-scale deployment of environmental sensors can lead to early warning systems; distributed monitoring of interdependent processes and control parameters in industrial plants ushers intelligent manufacturing solutions; and communication and collaboration between inter and intra-vehicular sensors can deliver intelligent transportation systems.

As a result, the number of edge devices is predicted to increase significantly over this decade, as shown in Figure 20.1. Interestingly, the problem of designing large-scale smart/intelligent systems has become tractable due to three factors:

a. Increased deployment of low-power data collection/sensing nodes;

b. Improved networking solutions via 4G/5G connectivity;

c. The advent of machine learning (ML) algorithms for ingesting large volumes of data to provide meaningful inference solutions.

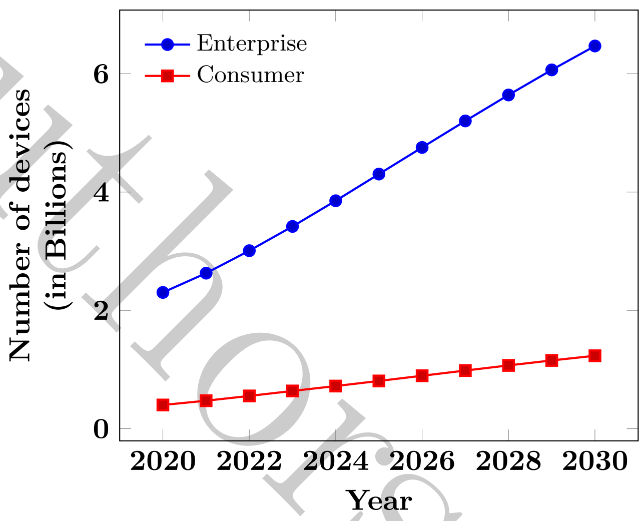

Figure 20.1: Predicted increase in edge devices from 2020 to 2030 in enterprise and consumer sectors [1].

Edge devices often serve as the sensor and actuators in an IoT ecosystem, whereas a central server provides the control decision. Therefore, increasing the number of connected edge devices results in better ML algorithms and precision control. Over the last decade, this successful scalable and *smart* system design approach (where edge nodes collect data and a central server processes this data using ML models for control decisions) has led to tremendous growth in the number of edge devices. This growth is expected to continue over the next decade resulting in more than 7 billion edge devices by 2030 [1].

Edge devices are primarily application-specific and designed with budget constraints in terms of power, area, and computation capacity. Hence, the security of these devices has often remained an afterthought in the system designing process. If adopted for critical infrastructure, this security-oblivious *smart* system design has the potential for severe consequences. For example, attackers can target PMUs in a smart grid via network-based attacks or GPS spoofing attacks to corrupt the phasor measurements and subsequently compromise the entire power system [2].

Unfortunately, weak and vulnerable edge devices are abundant in the current generation of intelligent IoT systems. Given the variety of edge devices and their multitudes of designs and operations, it is challenging to create targeted solutions for attack detection and countermeasure designs. Hence, it is an opportune time to introspect the progress made over the last few decades in attack detection and countermeasure development for the Internet of Things (IoT) edge and explore the open problems in this field. This chapter will present our exploration, discoveries, and introspection on the advances and opportunities of security research for edge resources.

20.2. Attack Surfaces for Edge Devices

An *attack surface* is defined as a pathway for compromising the integrity of the operation of a network-connected device. This pathway can originate from hardware, software, networking, or other system components connecting the device with the environment.

In edge devices, vulnerabilities originating from multiple surfaces can be exploited using different attack models and attacker goals. Hence, for each distinct attack, the type and intent of the attacker must be understood before designing countermeasures. In general, attack surfaces in edge devices are classified into three categories, *i.e.*, (1) physical/hardware, (2) software, and (3) network attack surfaces.

20.2.1. Physical Attack Surface

At the lowest level of system architecture, we have hardware that performs the computation and connects a computation's logic and control decisions to the physical world. Although traditional computer security primarily focuses on software vulnerabilities, increasing hardware exploitation in recent years has exposed the perils of hardware-oblivious security designs. Computation does not occur within a void; instead, it is realized through electronic signals and systems, and thus it leaves physical fingerprints. Therefore, hardware vulnerabilities arise from different attack surfaces, such as side and covert channels, leakage of execution time, fault injection, and the inclusion of malicious hardware components.

Hardware attacks become more prominent for edge devices due to the proximity and availability of the devices to the end user. For example, profiling-based side channel analysis requires data acquisition from similar devices. Thus, mimicking such attacks on large enterprise servers might cost attacker significantly, whereas profiling low-cost edge nodes are cheap and effective. Modern-day cryptography depends on Kerckhoff's principle, where security is inherently built on the secrecy of the cryptographic keys. Thus, using hardware side channels, an attacker can cost-effectively leak keys, and if such keys are used over the network, it becomes easier to infiltrate the network from the edge.

Moreover, attack oblivious circuit design opens up more straightforward yet effective attack surfaces. For example, firmware used in an edge device is often stored in flash memory. If the flash memory is not protected, an attacker can easily capture the contents using simple flash dump attacks [3]. Thus, attack surfaces on physical hardware in an edge device can have profound implications for the security of the entire IoT ecosystem.

20.2.2. Software Attack Surface

Software security for edge devices is also challenging due to the unique nature and application of these devices. Software attack surfaces on IoT edge devices arise from two distinct sources:

a. Insecure application code, and
b. Vulnerable operating system (OS).

Edge devices are generally budget constrained regarding computation power and memory. As a result, full-fledged operating systems are not the first choice while designing these systems. Therefore, designers use (1) bare-metal software, (2) custom-made OSes such as Mbed OS [4] and MicroBlaze [5], or (3) OSes tailored from existing full-fledged lightweight OS distributions such as Yocto [6], BuildRoot [7], and OpenWRT [8]. Interestingly, these design choices lead to different attack surfaces for the system.

Generally, bare-metal programming for embedded systems rarely considers secure coding practices and thus remains vulnerable to simple attacks such as buffer overflows, memory corruption, and code reuse [9]. In addition, security via obfuscation strategy in embedded system development still (falsely) provides acceptable security assumptions to the developers. As a result, embedded systems at the edge remain the most vulnerable to common software attacks.

On the other hand, system-specific OSes such as MbedOS and Microblaze suffer software vulnerabilities due to insecure implementation of applications, unpatched zero days, and relatively uncomplicated reverse engineering efforts. Finally, software in Linux-derived lightweight OSes enjoys better security and management options. However, insecure software writing practices still make the applications vulnerable to standard software attacks ranging from basic buffer overflows to complicated ROP attacks.

20.2.3. Network Attack Surface

The triumph of edge-device-centric intelligent system design has relied on the tremendous development in network connectivity required to maintain communication and control for many devices and systems. Less-regulated connectivity to many resource-constrained devices has the potential to create network security nightmares. This is evident in the increasing number of cyber attacks in the IoT ecosystem over the last few years, as shown in Figure 20.2.

Network attack surfaces are fundamentally attributed to weak passwords and default credentials, improper use of cryptography, poorly-defined access control, and lack of understanding of network security fundamentals for the edge. Assess control and resource management for a dynamic large-scale network is a complex problem, which can become highly challenging with resource-constrained devices connected to the network.

20.2.4. Goals of the Attacker

Attacks on edge devices aim at accessing the system to collect secret information, alter the system's runtime behavior, and disrupt or altogether turn off

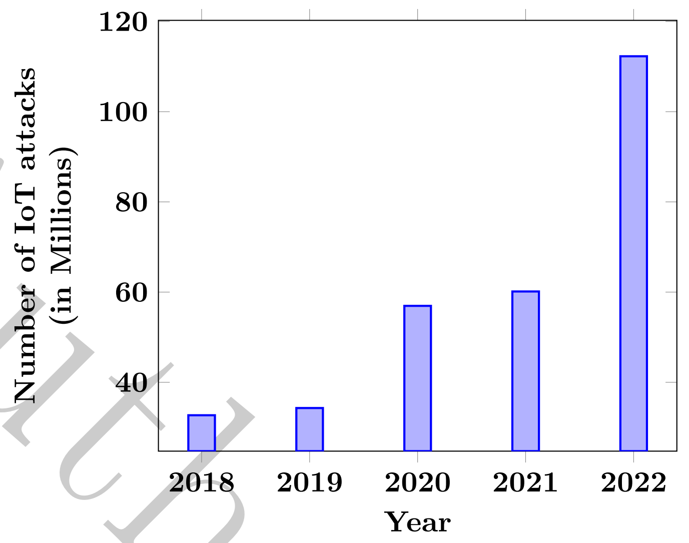

Figure 20.2: Number of IoT cyber attacks worldwide from 2018 to 2022 [10].

the system. We find that attacks at the edge nodes have some common goals:

G1. **Resource Hijacking.** Resource hijacking can be performed with software-based attacks. The attacker accesses the device first using software vulnerabilities and then, through privilege escalation, gains control over the system. Finally, the attacker manipulates the device to perform malicious tasks while keeping the process discreet from the victim. Common examples of such attacks are utilizing a device's resource for cryptocurrency mining, controlling a device as a botnet to launch a distributed denial of service (DDoS) attack, or using a device to spread spam and malicious software.

G2. **Information Leakage.** The goal of such an attack is to infer secret information (such as cryptographic keys, process algorithms, and user identity) from the device. Standard side channels and eavesdropping are examples of such attacks.

G3. **Subversion.** Subversive attacks usually target the underlying control algorithms in an IoT subsystem. For example, with the rise of machine learning-dependent large-scale controls, attackers can manipulate the sensor reports or network data from the edge to corrupt the control parameters or decisions.

G4. **Reverse Engineering.** In this scenario, the attacker aims to steal a device's firmware, design architecture, or sensitive intellectual property (IP). It is difficult to cluster reverse engineering (RE) activities as a covert or overt approach since the victim might be aware of such activities due to the reports of stolen devices or counterfeit products. However, the attacker usually aims to minimize the victim's awareness when such activities occur. In addition, RE efforts can also provide the necessary tools and efforts to perform an overt attack.

G5. **Device Function Disruption.** The goal of the attacker, in this case, is that the edge device will not perform as expected after the attack. Fault injection Attacks, such as bit-flip attacks, timing, and power glitches, can corrupt the running processes on a single device. Hence, the output behavior of the device will be unusual. When such attacks are launched over multiple devices, they can lead to service disruption and system failures.

G6. **Denial of Service.** This attack targets a complete or temporary device shutdown of a single or a group of devices. Networks level attacks, such as distributed denial-of-service (DDoS) attacks by Botnets and software-level attacks like malware attacks, can serve this purpose.

When building an intelligent system such as a smart grid or an edge robots-based automation solution, the designers should be aware of these common attack goals and surfaces for edge resources and develop their system accordingly with tighter access control and resource management policies.

20.3. Security Issues & Common Attacks in Edge Devices

20.3.1. Security Issues

Based on our discussions in this chapter so far, it is evident that edge devices present unique security challenges. We find that security issues in edge devices depend on factors such as outdated systems and lifecycle management issues, poor automation and control definition, weak privacy measures, and insecure scaling practices.

20.3.1.1. Outdated Systems

In the face of new and exciting products and systems launched each year, the lifecycle management of edge devices has become a challenging problem. Legacy devices coexist with newer ones in a network and can have severe security flaws due to the lack of support for end-of-life (EoL) devices. In addition, many legacy systems at the edge of IoT networks were not designed with security fundamentals at all.

Interestingly, competing financial motives can exist between the consumer and the manufacturer to replace or update existing devices. From an economic point of view, there is very little financial motivation for the manufacturers to continue support such as vulnerability patching, OS updating, and standard software maintenance. These activities require engineering resources, which from a business point of view, would be better utilized if used for new product development.

On the other hand, a user might continue using a functional device even after the manufacturer's support has ended for cost-saving reasons. These outdated devices are easily compromised using known vulnerabilities and zero-day attacks. The issue has become so severe that the FBI recently published an industry notification illustrating the cyber attack opportunities on unpatched and outdated medical devices and systems [11]. Thus, outdated EoL software, devices, and systems at the edge create significant security problems for the IoT ecosystem.

20.3.1.2. Weak Device and Network Management

Managing a network of heterogeneous devices is another complicated design issue for large-scale IoT networks, and this poor management can open up significant attack points to the network. For example, weak default configuration or exploitable side channel on the edge devices can efficiently be utilized to gain access to a system and attain the covert goals (*i.e.,* G1-G3) or the overt ones (*i.e.,* G5-G6).

Moreover, network management issues such as credential handling, access control, security monitoring, patching, updating, and resource management for the Internet of Things have yet to receive extensive evaluation and standardization processes. Therefore, manufacturers and users of edge devices seldom follow network hygiene for these systems.

As a result, common botnets employ simple tactics such as using the support passwords in edge devices to gain access to the network [12]. Thus, we have observed the success of botnets such as Mirai, which used a small list of (around 60) factory default credentials to hijack more than 300,000 IoT devices in 164 countries within a few months [13, 12]. From a security research point of view, it is clear that such exploitation will continue. As the easier "low-hanging" vulnerabilities (such as support passwords) are resolved, the attackers will deploy more complex attacks on network and device management tools to infiltrate IoT subsystems.

20.3.1.3. Privacy

Privacy is another critical concern in IoT edge devices. Since a significant subset of the devices is deployed in a user-facing environment, protecting user data becomes an imperative security problem to consider. The simplest solution remains to utilize cryptography for protecting user data and confidential information during transmission and storage. However, using cryptography in resource-constrained systems is complicated and sometimes prohibitively expensive. This leads to weaker resource protection and information leakage attacks at the edge. Moreover, end-to-end cryptography measures are difficult and costly to deploy and seldom have any financial incentive for the manufacturer.

20.3.1.4. Economics of Scale

Finally, scaling security fundamentals from the edge to the cloud remains a challenging problem. The unprecedented growth in the sheer volume of edge-based resources and cloud solutions for everyday issues has outpaced these systems' thorough security design and implementation. Therefore, scaling problems are rampant in large IoT ecosystems that support heterogeneous nodes and complex computing and control solutions. As a result, security has often been an afterthought in both enterprise and consumer space and has only been attended to when major security disasters emerge.

Table 1: Common malware found in IoT edge devices

Malware	Vulnerability Utilized	Common Usage
Mirai [12]	Default manufacturer credentials	DDoS
Remaiten [14]	Weak username and password	DoS, Malware Distribution
Linux.Wifatch [15]	Weak or default telnet credentials	Disconnecting a device
BASHLITE [16]	Common usernames and passwords	DDos, Creating C&C network
Linux.Darlloz [17]	CVE-2012-1823	Crypto mining
BrickerBot [15]	Weak telnet credentials	Destroying a device
Fusob [18]	User download	Mobile Ransomware
WannaCry [18]	CVE-2017-0144	Ransomware
Linux Spike Trojan (MrBlack) [19]	Default credentials	Man-in-the-middle, cookie hijacking
Stuxnet [20]	Zero days in Windows OS	Compromising SCADA systems

20.3.2. Common Attack Examples

Due to these existing security issues, attacks on edge devices have become increasingly common. These attacks can occur as a single type of attack or a combination of several types, depending on the attacker or the kind of attack. Table 1 provides a list of common malware found in IoT edge devices. A few of the well-known attacks are discussed below.

20.3.2.1. Malware for Distributed Denial of Service Attacks

With the increasing number of internet-connected edge nodes, one of the most trivial attacks is to launch distributed denial of service (DDoS) employing compromised edge devices. Some common malware used for such attacks are Mirai, Remaiten, and BASHLITE. Attackers usually scan over the internet for vulnerable edge devices and then infect them with malware that provides basic command and control capabilities, thus creating remotely controlled bots. Once the malware is installed over a large number of devices, the attacker performs a DDoS attack on a victim's website or service using the bots.

Botnet-based DDoS attacks require a command and control capability over many devices. It has been found that the attackers exploit the poor network hygiene of edge devices to build an army of botnets [12, 13, 14]. In most cases, using factory default credentials or common user-id and passwords lead to a small but potent dictionary that can compromise many devices. In addition, poor network management, over-generalized access control policies, and non-existing lifecycle maintenance of the devices, as discussed in the previous section, contribute to the success of botnet attacks.

20.3.2.2. Ransomware

A ransomware infects a computer system and encrypts critical data and resources with an attacker-provided encryption key. The victim can only retrieve the resource by obtaining the decryption key from the attacker by paying a significant ransom. Ransomware such as Fusob and WannaCry have demonstrated how mobile and cloud computing platforms can be compromised to gain ransom from a victim network [18]. Ransomware gains profitability and practical prominence due to the wide-scale use of cryptocurrencies which can provide complete anonymity over a financial transaction. Unprotected edge nodes can offer entry points for ransomware, and then poorly managed access control policies in the network lead to the compromise of critical resources.

20.3.2.3. Eavesdropping & Man in the Middle Attacks

Traditional eavesdropping and man-in-the-middle (MITM) attacks are mainly target specific and a part of advanced persistent threats (APTs). These attacks compromise the privacy guarantee of communication in the IoT network. Given the prominence of edge devices in the consumer space, the potential of private information leakage via eavesdropping and MITM attacks is alarming. For example, there have been reports on the hacking of home security cameras, baby monitors, and IP cameras via software vulnerabilities and common/weak credential usage [21].

20.3.2.4. Computer Resource Stealing Attacks

The advent of cryptocurrency mining and command and control (C&C) based DDoS attacks have made edge nodes attractive targets for resource stealing. Since many edge devices are automatically operated and maintained, resource-stealing malware remains unnoticed as long as there is no significant drop in performance. Thus, the victim ends up unknowingly paying for the computation cost. Since cryptocurrency mining and C&C operations provide lucrative opportunities, malware such as Mirai and Linux.Darlloz compete for device resources and try to retain control by eliminating other malware in a compromised edge device.

Moreover, some malware, such as Linux.Wifatch and Hajime do not steal the resources; instead, they disconnect devices from the network. Although they claim to be white-hat activists who remove vulnerable devices from the network, these attacks cause significant disruption to service.

20.3.2.5. Hardware Attacks

Low power and resource-constrained hardware in the edge devices are also targeted for hardware-based attacks. Hardware attacks are target specific and create entry points for other attacks. For example, side channel leakage of cryptographic keys used for encrypting a firmware of a given device lead to bot design or counterfeit manufacturing for that device. On the other hand, fault injection attacks help attackers bypass security measures. Hence, here we discuss these hardware attacks in detail.

Side Channel Analysis (SCA): Physical operation of edge devices can leak security-sensitive information through hardware side channels. In such cases, the attacker observes, extracts, and analyzes physical properties to infer information about cryptographic computation, execution details, and other critical functionalities. SCAs in edge devices are classified into two categories - (i) physical and (ii) non-physical.

- **Physical SCA:** During a physical side-channel attack, the attacker exploits the physical properties of the device, such as electromagnetic leakage, power consumption, and acoustic output. For example, power analysis attacks examine the instantaneous power consumption and use a statistical model (for power analysis during cryptographic computation) to infer information regarding the encryption key [22]. Similarly, acoustic attacks are performed by capturing and analyzing the acoustic waves generated by the chips during encryption using a microphone [23].
- **Non-Physical SCA:** In this attack, the attacker exploits a chip's non-physical parameters to collect the design credentials and the processes. For instance, during the timing attack, the attacker observes the routine runtime to obtain information about the running processes. In addition, cache-based

attacks monitor cache hit-and-miss timing differences to reveal system information [24].

Fault-Injection Attack: Unlike SCA, in a fault-injection (FI) attack, the attacker injects external faults into the device to modify the functionality, extract sensitive information, or disable the system. Some common FI attacks are discussed below.

- **Row Hammer Attack:** In this attack, bits in the DRAM cells are flipped, thus injecting faults into the device. DRAM cells consist of transistors and capacitors aligned in arrays. Due to the continuous scaling of DRAMs, the cell density is increasing; hence the electromagnetic coupling effect among the cells is increasing. It has been shown that activating the rows in a DRAM at a very high frequency disturbs the nearby cells. When the disturbance exceeds the threshold, bits in the nearby cells are flipped and corrupted [25].
- **Power Glitch Attack:** During the power glitch attack, the supply voltage is changed aggressively to modify the execution flow, specifically to skip the targeted instruction. This attack usually aims at bypassing security checks, avoiding the number of attempts barrier while launching a brute-force password break attack [26].
- **Clock Glitch Attack** This attack exploits the clock management capabilities in hardware. A *glitchy clock cycle* is a temporal voltage spike that can be generated by changing the clock source where both have the same clock frequency but slight phase differences. During the glitched clock signal, an invalid signal is received at the register, which corrupts the corresponding routine execution [27].

20.4. Attack Detection Techniques & Countermeasures

Detecting a new threat in an IoT ecosystem requires intelligent monitoring and surveillance services over the entire network. Additionally, hardware attacks can involve physical compromise or cloning of the devices, and therefore, situational and physical awareness of the network is also required to detect such attacks. Moreover, honeypots, zero-day management, and malware research also help discover attacks early. These topics are discussed in detail here.

20.4.1. Common Attack Detection Techniques

20.4.1.1. Real-time Monitoring and Honeypots

Real-time monitoring and analytic tools are standard in enterprise IoT networks. Different vendor products exist in the commercial domain that provides end-to-end IoT network monitoring solutions. General network attacks, *i.e.,* simple DDoSes, can be detected through these solutions. However, commercial solutions have several drawbacks, like cost, engineering efforts for installation and maintenance, generalized one-solution-for-all approach, and lack of compatibility among different solutions. Some of these drawbacks, such as compatibility issues and depth of protection, can be addressed through standardization efforts. On the other hand, other disadvantages, *(*e.g.,) detecting targeted attacks and protecting against such threats, will require a complete rethinking of network design and architecture for edge devices.

Honeypots provide active threat detection and analysis capabilities to the researchers. Thus, new malware that exploits zero days or performs weak credentials-based attacks can be detected through well-positioned honeypots at the edge [28]. Moreover, a well-engineered honeypot design that redirects or reroutes malware traffic from the network can provide active protection against DDoS attacks [29].

20.4.1.2. Machine Learning Tools

Large-scale attacks on IoT networks and edge devices start with abnormal behavior. Thus, effective anomaly detection techniques are imperative for the early diagnosis of a critical attack. Recent progress in data-oriented large-scale machine learning (ML) is poised to impact anomaly detection in the IoT network significantly. Interestingly, current progress in ML algorithms is mostly in supervised learning scenarios that are excellent in detecting patterns that they have learned from labeled training data. Thus, supervised learning-based models will have good precision and recall performance for attack signatures that the model has already learned. However, such an approach is only partially helpful in detecting novel attacks.

20.4.1.3. Physical Fingerprinting

For hardware-based attacks, real-time malware detection is possible through application signature monitoring [30, 31, 32]. Intra-processor side-channel analysis through shared memory systems or power lines can also provide real-time process monitoring capabilities to protect processors from running malicious or crypto-mining codes. In addition, the physical operation of edge devices and sensors leaves fingerprints in the collected data that can be leveraged to detect an attack on the node [33, 34].

20.4.2. Countermeasures

Active and passive countermeasures can secure edge resources and devices from future threats. Since vulnerability exploitation and malware propagation depends on common factors such as commonly used credentials and poor network management as discussed in Section 20.3, a well-informed system design with countermeasures will lead to a robust cyberspace. Endpoint protection, use of lightweight cryptography, proper manufacturer description criteria for secure configuration of edge devices, and security-conscious hardware designs will secure the next generation of consumer and enterprise edge resources.

20.4.2.1. Endpoint Authentication

Secure authentication protocols that do not use factory default credentials and do not allow commonly used passwords provide security against botnet generation. However, designing automated multi-factor authentication protocols for the edge nodes seldom accessed by operators or users is challenging. For these scenarios, carefully designed protocols and hardware-based security primitives such as physical unclonable functions (PUFs), fingerprints, and root of trusts (*i.e.,* trust zones and secure enclaves) can provide better authentication guarantees at the edge [35, 36, 37].

20.4.2.2. Lightweight Cryptography

Cryptographic protocols are mandatory on data during transit and at rest for data privacy and resource protection at the edge nodes. Unfortunately, standard cryptographic algorithms such as Advanced Encryption Systems (AES) for private key and RSA algorithms for public key cryptography demand additional resources from budget-constrained devices. Fortunately, there have been significant advances in lightweight cryptography for small devices in recent years. Recently (February 2023), NIST selected Ascon – a family of cryptographic algorithms for low-power resource-constrained devices [38]. This standardization of lightweight cryptography promises to solve the long-existing standard encryption and key management problems for IoT networks.

20.4.2.3. Manufacturer Usage Descriptions (MUDs)

Unauthorized malicious devices can enter a network by masquerading or cloning harmless edge nodes. Such attacks use ambiguity in the network usage permission of the edge resources. To thwart such exploitation, NIST has recently provided guidelines for manufacturer usage descriptions (MUDs) [39]. MUDs are manufacturer-defined resource usage descriptions that are assigned to individual edge devices. When connected to a new network, a MUD-supported device provides details of the access requirements for its operation. The manufacturer predetermines a given device's network use and resource access. This enables stricter access control policies without compromising the activities of trusted nodes. This strategy also moves part of the security burden to the manufacturer of edge products.

20.4.2.4. Zero Trust Architecture

Recent cyber attacks on networked components leverage the lack of granular access control of resources. For example, attackers first target edge components to access the network and then perform privilege escalation to exploit a critical resource that lacks proper access control mechanisms. This way, data and other resources such as computation, sensor readings, and actuator control can be stolen or exploited via weak and outdated access control policies for shared resources.

To thwart the vulnerabilities arising from traditional enterprise firewalls that use generalized and broad access control policies, zero trust design for network architecture has been proposed [40, 41, 42, 43]. Zero trust networks fundamentally differ from traditional static, broad access controlled designs and employ detailed and dynamic resource monitoring and access policies. The core concepts of zero trust architecture revolve around granular access control, least privilege for resource utilization, dynamic trust validation, and smaller *trust zones*. To provide a well-defined standard framework for zero trust architecture, NIST has recently (August 2020) published a zero trust architecture guideline that provides the necessary details an enterprise network should maintain to design a zero trust solution [41].

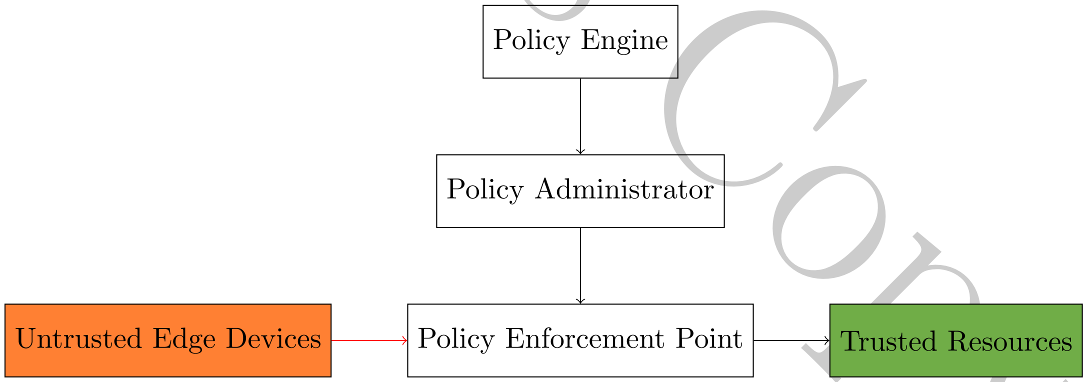

Figure 20.3: Basic components of a zero trust architecture.

The key logical components of a zero-trust architecture are the policy engine, policy administrators, and policy enforcement points, as shown in Figure 20.3. In a zero-trust network, resources are defined with an all-inclusive approach. Hence, not only sensitive data marked for tighter access, relatively standard network components, *i.e.*, computing resources, trusted edge connected devices, and verified low-power sensors and actuators are all tagged as trusted resources in the network. Access to these trusted resources is rigorously maintained using the key components. The policy engine provides the ultimate access decision (*i.e.,* grant, deny, and revoke) for a networked resource to an incoming or existing device/node in the network. The policy engine utilizes standard access policies and real-time threat intelligence data to execute dynamic trust algorithms, ensuring that only trusted entities can access the resource [41].

Zero trust designs enforce that *every* resource access be monitored and controlled by the policy enforcement point, and thus move away from the *lazy* and implicit trust solutions that allow open (resource) access to previously trusted entities. This results in a vigilant networking solution that does not inherently trust any given entity even if it was authenticated before (hence the term zero trust). With the broader definition of resource, this zero trust mechanism overseen by a dynamic policy engine and enforced by the enforcement points delivers smaller but effective trust zones at the edge. Hence, zero trust networks are promising for securing the smart power grid solutions and other critical infrastructures modernized by edge-based technologies.

20.5. Conclusions and Future Research Directions

Scalable security solutions are imperative for the ever-increasing number of edge resources and devices in the IoT ecosystem. This chapter discusses the common attacks that have exploited edge devices and the existing measures to detect and counter these threats. Over the last decade, simpler exploits such as factory default passwords were practical enough for global-scale cyber-attacks. Fortunately, the situation has improved. However, existing vulnerabilities from the hardware, software, and network layers have the potential to compromise edge devices for the next large-scale cyber attack. Hence, significant research and standardization efforts should be pursued for usable security at the IoT network edge.

Newer solutions like zero trust architecture and MUD-based access control policies offer more granular and application-specific security solutions. Unfortunately, deploying these protocols require additional resources and support. Therefore, secure transitioning protocols to finer access control should be investigated, which will create an on-ramp for the existing/legacy network.

Early detection of an active threat can significantly reduce the damage. Therefore, novel monitoring and detection techniques should be investigated. Edge nodes in a network usually have predictable behavior, network signature, and hardware fingerprint. These properties need to be carefully fused for robust monitoring and anomaly detection. Research and development initiatives in unsupervised and reinforcement learning algorithms for characterizing and isolating novel threats need to be pursued by the academic community and industry. Translating the revolution in ML for cyber security problems would create a paradigm shift in secure and robust network design.

In addition, proper deployment of cryptography for resolving privacy issues requires carefully engineered effort. The recent developments in lightweight cryptography will usher in a plethora of products and encryption solutions for the edge. However, their integration of the legacy system will be an exciting research avenue. Other advanced primitives, *e.g.,* fully homomorphic encryption algorithms, offer cloud and edge computation on encrypted data. These primitives are enjoying rapid development and are expected to impact the data security in edge devices in the near future.

Finally, hardware security problems in edge devices are under-focused and remain open threats to the safety of future IoT networks. Hardware attacks can be very target-oriented and cause entry points for firmware reverse engineering, data breach, and authentication compromises. Interestingly, hardware-based authentication and security solutions offer more robust entity management and guarantees. In each passing quarter, more security enclaves, trust-zone-supported microprocessors, and microcontrollers enter the edge device market. This hardwired root-of-trust at the device level has the potential to enable robust trust propagation solutions from the edge to the cloud. Therefore, new research initiatives are required to transcend hardware from the weakest to the strongest link in IoT security.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. 2245156. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Bibliography

[1] Transforma Insights. Number of edge enabled internet of things (iot) devices worldwide from 2020 to 2030, by market. https://www.statista.com/statistics/1259878/edge-enabled-iot-device-market-worldwide/. Accessed:August 10, 2023.

[2] Md Tanvir Arafin, Dhananjay Anand, and Gang Qu. A low-cost gps spoofing detector design for internet of things (iot) applications. In *Proceedings of the on Great Lakes Symposium on VLSI 2017*, pages 161–166, 2017.

[3] Mauricio Tellez, Samy El-Tawab, and M Hossain Heydari. Iot security attacks using reverse engineering methods on wsn applications. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pages 182–187. IEEE, 2016.

[4] João Alves. Arm mbed os 5.10 release: Focus on connectivity, firmware management and ease of use. 2018.

[5] Pong P Chu. *FPGA Prototyping by SystemVerilog Examples: Xilinx MicroBlaze MCS SoC Edition.* John Wiley & Sons, 2018.

[6] Otavio Salvador and Daiane Angolini. *Embedded Linux Development with Yocto Project.* Packt Publishing Ltd, 2014.

[7] Thomas Petazzoni and Free Electrons. Buildroot: a nice, simple and efficient embedded linux build system. In *Embedded Linux System Conference*, volume 2012, 2012.

[8] Florian Fainelli. The openwrt embedded development framework. In *Proceedings of the Free and Open Source Software Developers European Meeting*, page 106, 2008.

[9] Abraham A Clements, Naif Saleh Almakhdhub, Khaled S Saab, Prashast Srivastava, Jinkyu Koo, Saurabh Bagchi, and Mathias Payer. Protecting bare-metal embedded systems with privilege overlays. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 289–303. IEEE, 2017.

[10] SonicWall. Annual number of internet of things (iot) malware attacks worldwide from 2018 to 2022 (in millions) [graph], in statista. https://www.statista.com/statistics/1377569/worldwide-annual-internet-of-things-attacks/. Accessed: August 12, 2023.

[11] FBI Cyber Division. Private industry notification: Unpatched and outdated medical devices provide cyber attack opportunities. https://www.ic3.gov/Media/News/2022/220912.pdf. Accessed: August 12, 2023.

[12] Graham Cluley. These 60 dumb passwords can hijack over 500,000 iot devices into the mirai botnet. https://grahamcluley.com/mirai-botnet-password/. Accessed: August 12, 2023.

[13] Jane Devry. Mirai botnet infects devices in 164 countries. https://www.cybersecurity-insiders.com/mirai-botnet-infects-devices-in-164-countries/. Accessed: August 12, 2023.

[14] M Shobana and S Rathi. Iot malware: An analysis of iot device hijacking. *International Journal of Scientific Research in Computer Science, Computer Engineering, and Information Technology*, 3(5):2456–3307, 2018.

[15] Michele De Donno, Nicola Dragoni, Alberto Giaretta, and Manuel Mazzara. Antibiotic: protecting iot devices against ddos attacks. In *Proceed-*

ings of 5th International Conference in Software Engineering for Defence Applications: SEDA 2016 5, pages 59–72. Springer, 2018.

[16] Artur Marzano, David Alexander, Osvaldo Fonseca, Elverton Fazzion, Cristine Hoepers, Klaus Steding-Jessen, Marcelo HPC Chaves, Ítalo Cunha, Dorgival Guedes, and Wagner Meira. The evolution of bashlite and mirai iot botnets. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 00813–00818. IEEE, 2018.

[17] Vipindev Adat and BB Gupta. A ddos attack mitigation framework for internet of things. In *2017 international conference on communication and signal processing (ICCSP)*, pages 2036–2041. IEEE, 2017.

[18] Chris Adams. Learning the lessons of wannacry. *Computer Fraud & Security*, 2018(9):6–9, 2018.

[19] Igal Zeifman, Ofer Gayer, and Ronen Atias. Lax security opens the door for mass-scale abuse of soho routers. https://www.incapsula.com/blog/ddos-botnet-soho-router.html. Accessed: August 12, 2023.

[20] Ralph Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy*, 9(3):49–51, 2011.

[21] Daniel Wroclawski. How to keep your home security cameras from being hacked. https://www.consumerreports.org/home-garden/home-security-cameras/keep-home-security-cameras-from-being-hacked-a2927068390//. Accessed: August 12, 2023.

[22] Thanh-Ha Le, Cécile Canovas, and Jessy Clédiere. An overview of side

channel analysis attacks. In *Proceedings of the 2008 ACM symposium on Information, computer and communications security*, pages 33–43, 2008.

[23] Daniel Genkin, Mihir Pattani, Roei Schuster, and Eran Tromer. Synesthesia: Detecting screen content via remote acoustic side channels. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 853–869. IEEE, 2019.

[24] Goran Doychev, Boris Köpf, Laurent Mauborgne, and Jan Reineke. Cacheaudit: A tool for the static analysis of cache side channels. *ACM Transactions on information and system security (TISSEC)*, 18(1):1–32, 2015.

[25] Onur Mutlu and Jeremie S Kim. Rowhammer: A retrospective. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(8):1555–1571, 2019.

[26] Kamil Gomina, Jean-Baptiste Rigaud, Philippe Gendrier, Philippe Candelier, and Assia Tria. Power supply glitch attacks: Design and evaluation of detection circuits. In *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 136–141. IEEE, 2014.

[27] Johannes Obermaier, Robert Specht, and Georg Sigl. Fuzzy-glitch: A practical ring oscillator based clock glitch attack. In *2017 International Conference on Applied Electronics (AE)*, pages 1–6. IEEE, 2017.

[28] Mohamad Faiz Razali, Muhammad Nazim Razali, Fawwaz Zamir Mansor, Gopinath Muruti, and Norziana Jamil. Iot honeypot: A review from researcher's perspective. In *2018 IEEE Conference on Application, Information and Network Security (AINS)*, pages 93–98. IEEE, 2018.

[29] M Anirudh, S Arul Thileeban, and Daniel Jeswin Nallathambi. Use of honeypots for mitigating dos attacks targeted on iot networks. In *2017 International conference on computer, communication and signal processing (ICCCSP)*, pages 1–4. IEEE, 2017.

[30] Mohsen Damshenas, Ali Dehghantanha, Kim-Kwang Raymond Choo, and Ramlan Mahmud. M0droid: An android behavioral-based malware detection model. *Journal of Information Privacy and Security*, 11(3):141–157, 2015.

[31] Takamasa Isohara, Keisuke Takemori, and Ayumu Kubota. Kernel-based behavior analysis for android malware detection. In *2011 seventh international conference on computational intelligence and security*, pages 1011–1015. IEEE, 2011.

[32] Parvez Faruki, Ammar Bharmal, Vijay Laxmi, Vijay Ganmoor, Manoj Singh Gaur, Mauro Conti, and Muttukrishnan Rajarajan. Android security: a survey of issues, malware penetration, and defenses. *IEEE communications surveys & tutorials*, 17(2):998–1022, 2014.

[33] Md Tanvir Arafin and Kevin Kornegay. Attack detection and countermeasures for autonomous navigation. In *2021 55th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6, 2021. doi: 10.1109/CISS50987.2021.9400224.

[34] Tsion Yimer, Md Tanvir Arafin, and Kevin Kornegay. Securing industrial control systems using physical device fingerprinting. In *2020 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pages 1–6, 2020. doi: 10.1109/IOTSMS52051.2020.9340160.

[35] Md Tanvir Arafin. *Hardware-Based Authentication for the Internet of Things*. PhD thesis, University of Maryland, College Park, 2018.

[36] Mingze Gao, Qian Wang, Md Tanvir Arafin, Yongqiang Lyu, and Gang Qu. Approximate computing for low power and security in the internet of things. *Computer*, 50(6):27–34, 2017.

[37] Jiliang Zhang, Chaoqun Shen, Haihan Su, Md Tanvir Arafin, and Gang Qu. Voltage over-scaling-based lightweight authentication for iot security. *IEEE Transactions on Computers*, 71(2):323–336, 2022. doi: 10.1109/TC.2021.3049543.

[38] NIST selects 'lightweight cryptography' algorithms to protect small devices. https://www.nist.gov/news-events/news/2023/02/nist-selects-lightweight-cryptography-algorithms-protect-small-devices. Accessed: August 12, 2023.

[39] Eliot Lear, Ralph Droms, and Dan Romascanu. Manufacturer usage description specification. Technical report, 2019.

[40] Yuanhang He, Daochao Huang, Lei Chen, Yi Ni, and Xiangjie Ma. A survey on zero trust architecture: Challenges and future trends. *Wireless Communications and Mobile Computing*, 2022, 2022.

[41] Scott et. al. Rose. Zero trust architecture. *NIST special publication*, 800:207, 2020.

[42] Elisa Bertino. Zero trust architecture: does it help? *IEEE Security & Privacy*, 19(05):95–96, 2021.

[43] Naeem Firdous Syed, Syed W Shah, Arash Shaghaghi, Adnan Anwar,

Zubair Baig, and Robin Doss. Zero trust architecture (zta): A comprehensive survey. *IEEE Access*, 10:57143–57179, 2022.