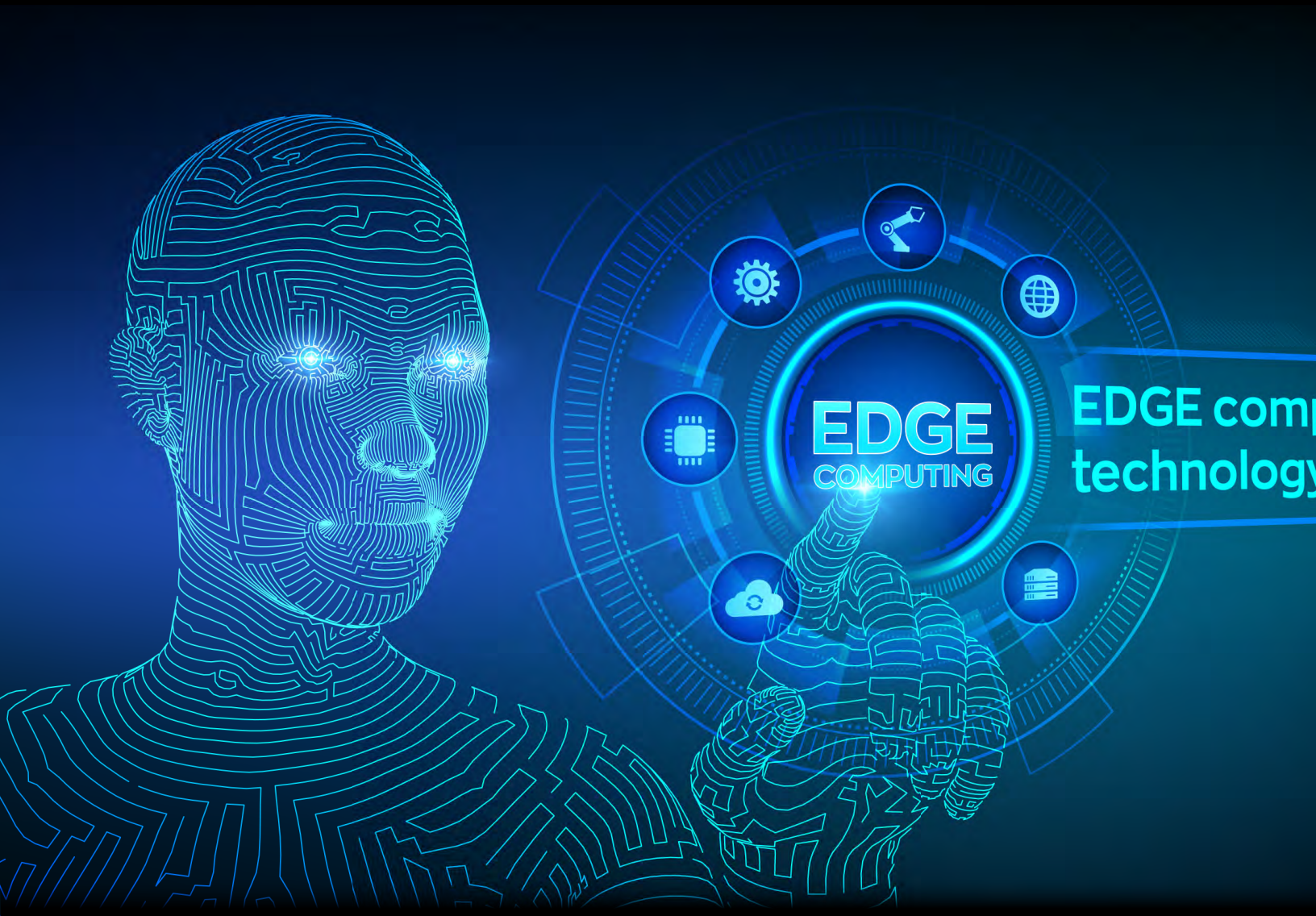
GlobalLogic®



Immunizing Edge Computing

by Ashish Garg and Bipin Gupta

Contents

Introduction

Edge computing refers to taking the computation execution resources (computing and storage) out of the traditional data center and bringing them as close as possible to the location where they are needed. They can be moved to hand-held devices, appliances, or physical units, typically within or at the boundary (edge) of the access networks. It provides low latency, high bandwidth, and more secure computing and storage.

In today's world, applications running on devices are getting more sophisticated and smarter every day. There are many opportunities in edge computing for both end consumers and industries. Edge-enabled applications are capable of providing a seamless and personalized experience to end users, help companies adapt to the dynamically changing needs of their industries, and much more.

The aim is to harness the processing power of end devices and nodes like routers, switches, etc., to perform data computation in order to reduce the amount of data that is transferred to a cloud server over a network.

Edge computing has lots of potential and shall be considered from the perspective of enterprise where it can be a next-generation solution for all upcoming industry use cases like the Internet of Things (IoT) and the automotive and telecom sectors. The edge computing landscape is distributed widely and is evolving every day. However, industrial standards and business models are not set up for edge computing. Several big players need to be involved to create end-to-end solutions so that any enterprise can expand offerings in their industry.

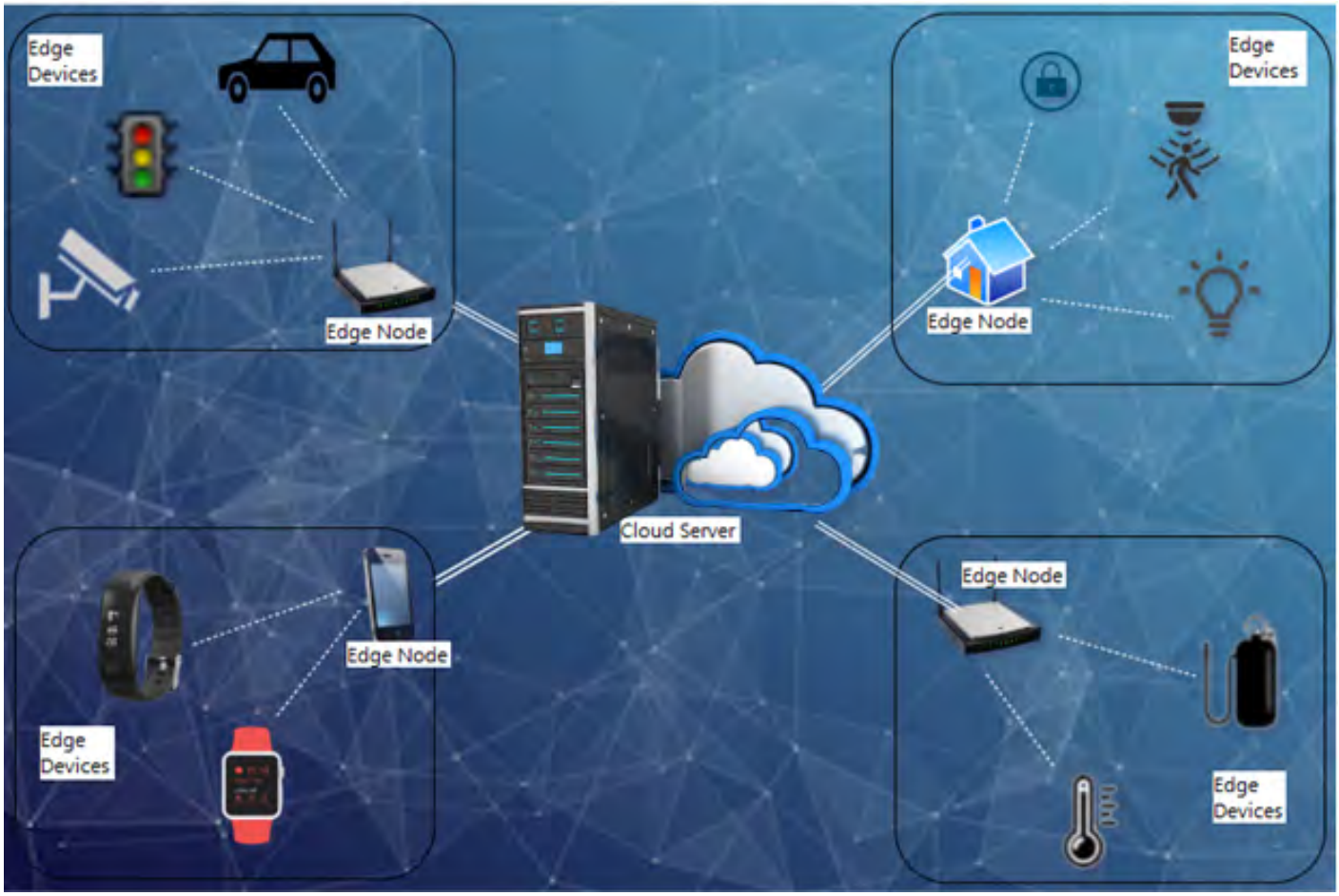
This white paper describes edge computing in detail, its meaning to different industries, the associated security challenges, and explores the possible defense mechanisms needed to secure it against those challenges. This information can be used to choose a suitable defense model against security attacks, depending on the use case of each industry.

The Edge Computing Landscape

An Edge computing ecosystem consists of a large number of systems like hardware vendors, platform companies, application developers, IoT devices, edge devices, and telecom providers. Apart from these, there are two major key players in the ecosystem, i.e., Hyperscale Cloud Providers (HCPs) and Operations Technology (OT) vendors. Amazon Web Services (AWS), Microsoft Azure, and Google are a few HCPs that provide core cloud infrastructure and platforms. They have thousands of developers working towards building application ecosystems that can serve multiple enterprises in many sectors globally.

It is a distributed landscape where the computing responsibilities are being shared between edge devices and the central cloud server, thus reducing the load on cloud servers. Edge devices processes some of the data locally and transmits only relevant information to the cloud server for further and complex processing. This way, edge computing helps to achieve faster responses to the end device/user, low bandwidth utilization, etc.

An example of the edge computing landscape has been illustrated in the figure below.



Application of Edge Computing

Telecom

Edge Computing in telecom is often referred to as Mobile Edge Computing (MEC) or Multi-Access Edge Computing. Again, these terms have a similar meaning: execution resources (computation and storage) are moved to applications with networking close to the end users, typically within or at the boundary of the operator networks.

Mobile Edge Computing is pivotal to the 5G platform and allows telecom providers to capture new opportunities. It is predicted that 5G will account for one-fifth of all mobile data traffic by 2023, where 25% of the use-cases may depend on edge computing capabilities.

A significant portion of the potential 5G revenue is expected to come from enterprise and IoT services, which will rely heavily on edge computing architecture. Therefore, the edge capabilities of each node in the network will be a primary focus of a 5G infrastructure for any service provider.

Automotive & Manufacturing

In the automotive and manufacturing industries, there is a huge demand for smart applications capable of faster processing of data, resulting in higher accuracy and increased productivity. Such applications would need to process heavy loads of real-time data. It is believed that distributed edge computing architecture is a key technology necessary to support such use cases.

The automotive industry is working towards making driving safer, increasing the flow of traffic, and designing vehicles to consume energy more efficiently with lower emissions.

Automated and intelligent driving, the creation and distribution of advanced maps with the real-time data, and advanced driving assistance using cloud-based analytics of UL (Up Link) video streams are a few emerging use cases that require vehicles to be connected to the cloud.

IoT (Internet of Things)

Almost all kinds of electrical devices will eventually become part of IoT. Devices such as air quality sensors, LED bars, streetlights, and even internet-connected microwave ovens both produce and consume data.

The International Data Corporation predicts that 41.6 billion devices will be connected to the Internet by 2025 [1]. Some of them might require very quick response times, some might involve sensitive information, and some may generate large amounts of data that might demand high network bandwidth. Cloud computing will not be efficient enough to support all these needs, so data will need to be consumed at the edge of the network itself.

With edge computing, the enormous amount of processing required can be broken into different levels. Edge devices will perform the initial computation and filtering or limiting of data that actually needs to be transmitted to the backend cloud servers, thus reducing the network traffic and response time for smaller and simple needs.

Security Threats to Edge Computing

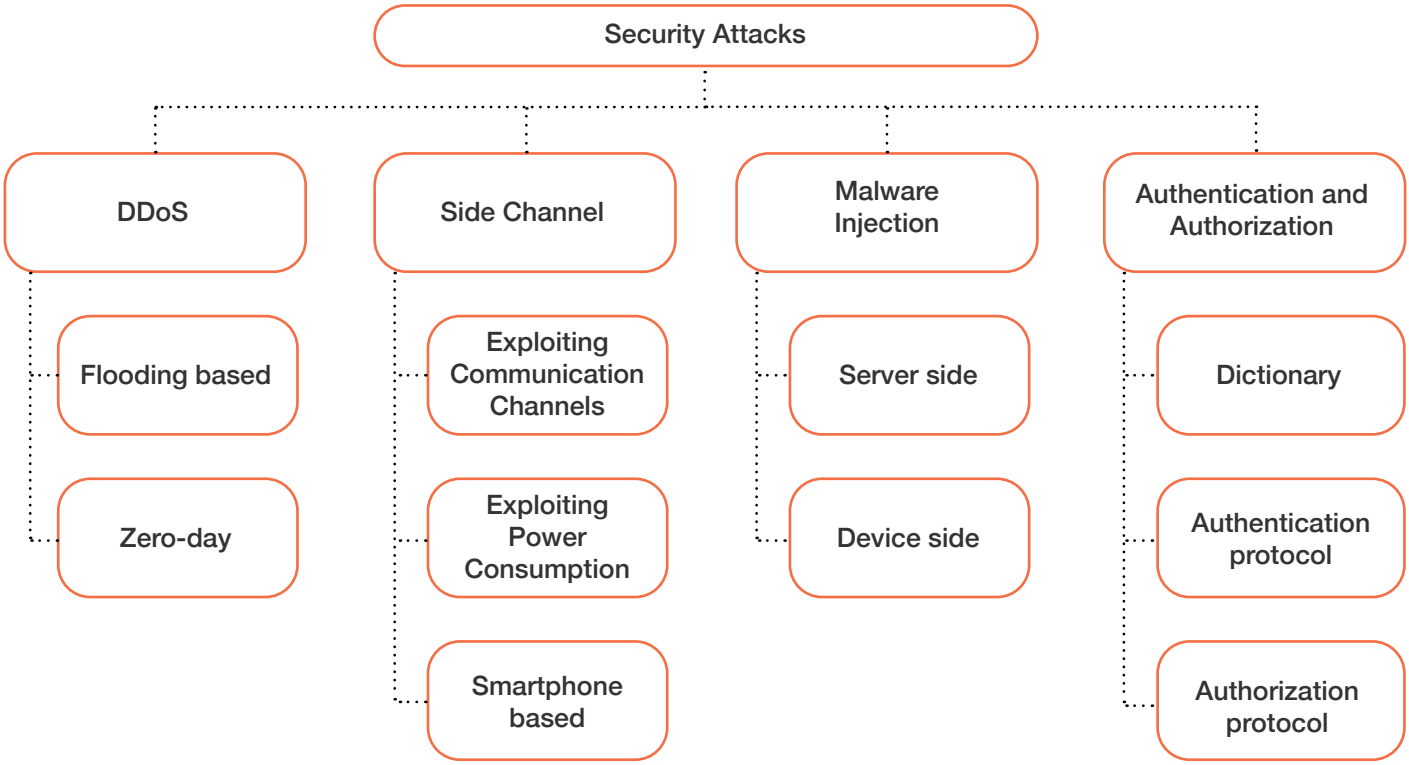
In this section, we will first summarize the major security threats in edge computing. These threats have mainly resulted from design flaws, misconfigurations, and implementation bugs. Then, we will elaborate on corresponding defense mechanisms that are either detection- or prevention-based with the intent of preventing attacks from occurring. Finally, we will outline the root causes of each type of attack and explore the practicality of defending against them.

A weak authentication mechanism is one of the most frequent vulnerabilities. For example, a WiFi networks security breach. Among the 439 million households who use wireless connections, 49% of WiFi networks are unsecured, and 80% of households still have their routers set on default passwords. For public WiFi hotspots, 89% of them are unsecured, as explained in reference [2].

As we move towards safeguarding the edge landscape, we need to build awareness of privacy and security threats. All stakeholders, including service providers, system and application developers, and end users need to be aware that the user's privacy can be disrespected without notice at the edge of the network.

Edge computing threats can be segregated into four broader categories, as shown below. In order to immunize edge computing, we first need to understand them and prepare a holistic approach to tackle each one of them.

Types of Security Threats



1. DDoS attacks

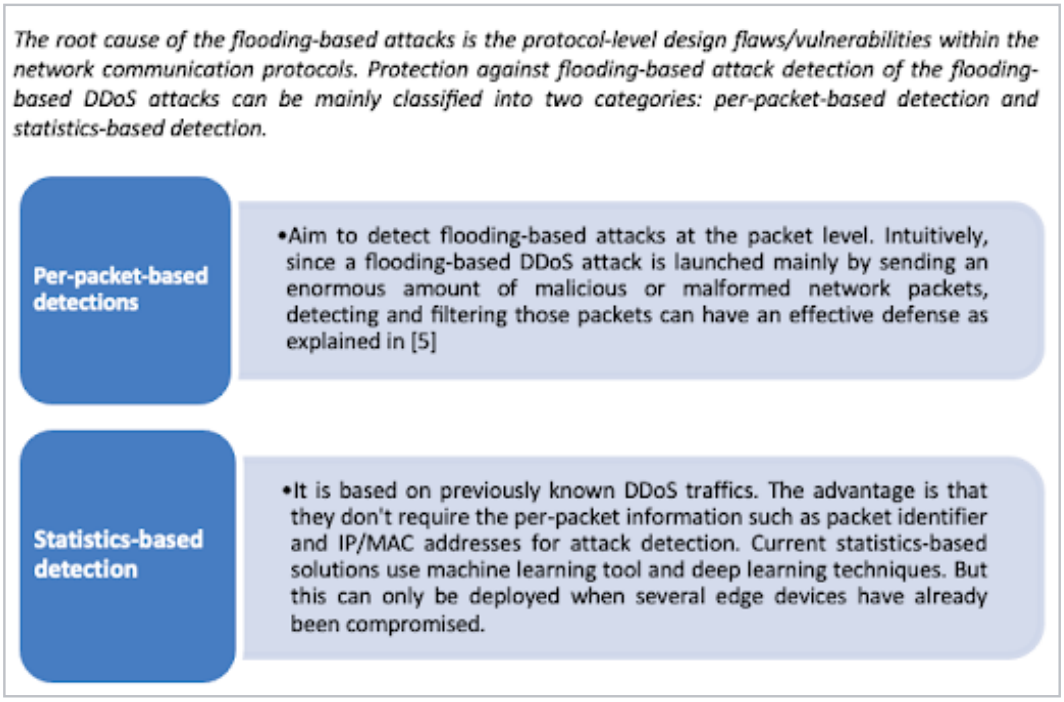
A distributed denial-of-service (DDoS) attack is a malicious attempt to disrupt the normal traffic of a targeted server, service or network. It can be classified further into two types: Flooding-based attacks and Zero-day attacks.

a. Flooding-based attacks

This is a type of DDoS attack aimed at shutting down the normal service of a server based on a large amount of flooded malformed or malicious network packets.

Depending upon the attacking techniques, it can be further categorized as UDP flooding, ICMP flooding, SYN flooding, ping of death (PoD), HTTP flooding, or Slowloris.

Possible Solutions:



The root cause of flooding-based attacks is the protocol-level design flaws or vulnerabilities within the network communication protocols. Protection against flooding-based DDoS attacks can be classified into two main categories: per-packet-based detection and statistics-based detection.

Per-packet-based detection

- Aims to detect flooding-based attacks at the packet level. Intuitively, since a flooding-based DDoS attack is launched mainly to send an enormous amount of malicious or malformed network packets, detecting and filtering those packets can be an effective defense, as explained in reference [5].

Statistics-based detection

- This is based on previously known DDoS traffic. The advantage is that they don't require the per-packet information such as packet identifier and IP/MAC addresses for attack detection. Current statistics-based solutions use machine learning tools and deep learning techniques. But this can only be deployed when several edge devices have already been compromised.

b. Zero-day attacks

A zero-day DDOS is more difficult to achieve and requires technical expertise. First, an unknown vulnerability (called a zero-day vulnerability) has to be found in a piece of code running on the target edge server/device, which can cause memory corruption and result in service shutdown.

One of the most common vulnerabilities and exposures is a heap-based overflow (CVE)-2010-3972, which can cause a DoS on Internet Information Services (IIS) 7.0 and IIS 7.5, as explained in reference [3].

Possible Solutions:

The root cause of zero-day attacks lies in the code-level vulnerabilities that can trigger memory failures/corruptions, which is difficult to spot, especially with edge devices due to the unavailability of their source code.

One possibility is to analyze the firmware. There have been many studies conducted in the past that have performed memory analysis based only on firmware. A few studies, like in references [4] and [5], have shown that with the help of deep learning models like recurrent neural networks (RNNs), graph neural networks (GNNs), and deep natural language processing (NLP), the vulnerabilities in firmware can be identified with better accuracy.

Another solution, as proposed in reference [6], is an in-process memory isolation extension module to the binary. However, it requires high computing resources, making it less feasible for IoT devices that are already resource-constrained.

Another study from reference [7] showed that building an IoT firewall using software-defined networking (SDN) can reduce the attack surface of an exposed IoT device.

2. Side-channel attacks

Side-channel attacks are when attackers use any publicly accessible information that is not privacy-sensitive in nature (side-channel information) to bypass a user's security and privacy.

Such public information is typically correlated "secretly" with certain privacy-sensitive data that should have been protected. Attackers can explore the hidden correlations to infer the protected data from the side channels. Since any public information can potentially link to sensitive data, side-channel attacks can happen anywhere in the edge computing landscape.

The architecture of a typical side-channel attack is when an attacker constantly obtains certain side-channel information from the target edge computing device and then feeds it into specific algorithms or machine-learning models that outputs the desired sensitive information. The most popular side channels in edge computing include communication signals, electric power consumption, and smartphone /proc filesystem or embedded sensors.

Attacks exploiting communication channels

In edge computing, exploiting communication signals transmitted between two edge nodes has a high potential of finding any sensitive information due to rich channel information. In this case, an attacker could be a malicious node itself, which might not be an edge device or an edge server, but which continuously sniffs the network traces and tries to extract sensitive information.

Attacks exploiting power consumption

Power consumption is an indicator of the electric usage of a system. It carries information related to either the device that consumes the energy (because different devices have different power consumption profiles when operating) or the intensity of computations in a computing task. Hence, the attackers might exploit its link to any sensitive data. We further categorize these types of attacks into two subclasses: attacks exploiting power consumption collected by meters and those exploiting power consumption collected by oscilloscopes.

Attacks exploiting smartphone-based channels

Smartphones are key edge devices in many applications. Smartphones have more advanced OSes and possess richer system information as compared to IoT devices. Hence, smartphones expose a broader attack surface as compared to less advanced IoT devices. These attacks can be further classified into two types: attacks exploiting the /proc filesystem and attacks exploiting the embedded sensors in smartphones.

The '/proc' is a system-level file system created by the kernel in Linux. It contains sensitive system information like interrupt and network data. Even though it is a system-level file, it is readable by the user-level threads and applications. Hence, accessing the /proc filesystem does not require any additional permission. Due to this, '/proc' is widely exploited to perform side-channel attacks.

Possible Solutions:

The root cause of any side-channel attacks is the hidden correlation, which could be very complicated and hard to identify, between the sensitive data to be protected and the publicly available side-channel information. Apparently, defenses against side-channel attacks can be performed from two directions: restricting access to side-channel information and protecting the sensitive data from inference attacks.

Obviously, there exists no feasible defense mechanism that can restrict access to uncontrollable side channels, leaving sensitive data protection the only approach for such attacks. In this section, we will first put forward an overview on data perturbation, a well-researched technique for protecting sensitive data from inference attacks. Then, we will summarize the defense mechanisms that can restrict access to side-channel information.

Data perturbation

The most well-known perturbation algorithm to protect sensitive data from inference attacks is k-anonymity, which modifies the identifier information of a piece of data before publishing its sensitive attributes, making it distinguishable from another k-1 piece of data. These k pieces of data form an equivalence class. Machanavajjhala et al. found that k-anonymity suffers from the homogeneity attack when the values of a sensitive attribute within an equivalence class are identical. To overcome this issue, they proposed l-diversity by ensuring each equivalence class has at least l distinct values for each sensitive attribute.

Restricting access to side channels

An alternative approach to defend against side-channel attacks is to restrict access to the side channel information by Obfuscation of Side-channel on the source code level. Molnar et al. [12] proposed a mechanism to eliminate control-flow-side-channel attacks from the C source code. Zhang et al. [13] developed a side channel detection scheme to monitor the abnormal cache behaviors on cloud and edge servers. These two methods directly perturb the side channels to obstruct the accuracy of inference algorithms used for side-channel attacks.

Possible Solutions:

Future research on defense against side-channel attacks in edge computing may focus on enhancing access control models to better regulate the access to the controllable side channels and the published data. Also, besides being used as attack resources, side channels can also be used as defense resources.

The research conducted by Clark et al. [8] demonstrated a good way to detect malware based on power consumption. Hence, we believe that using side channels to strengthen defense mechanisms may be a good candidate for future research.

3. Malware injection attacks

The action of effectively and stealthily injecting/installing malware into a computing system is referred to as a malware injection attack. This is one of the most dangerous attacks since malware is a big threat to system security and data integrity. In the traditional internet or general-purpose computer infrastructures where strong computational power is available to support high-performance firewalls or other threat protection systems, malware injection is not always feasible or possible. However, edge devices and the low-level edge servers can barely be protected by a traditional firewall and hence are more vulnerable to malware injection attacks, which can be categorized as follows:

a) Server-side injections

There are four main types of injection attacks targeting edge servers: SQL injection, cross-site scripting (XSS), Cross-Site Request Forgery (CSRF) and Server-Side Request Forgery (SSRF), and Extensible Markup Language (XML) signature wrapping.

SQL injection is a code injection technique that destroys the backend databases. To construct a normal SQL query, a legitimate user is allowed to manipulate only the designated areas (e.g., name and date) to get results from the server. However, an attacker may circumvent this constraint by inputting escape characters along with the query string. In this case, the server may mistakenly execute everything the attacker inputs after the escape characters.

This vulnerability usually exists when a database management system does not filter escape characters for SQL processing. SQL injection is not only a serious threat to data confidentiality and integrity but also allows attackers to inject malicious scripts.

XSS is a client-side attack in which an attacker injects malicious codes (usually HTML/JavaScript codes) into data content, which can be accessed and executed automatically by the servers.

Possible Solutions:

Defenses against server-side injections also consider the four major types of attacks: SQL injection, XSS, CSRF/SSRF, and XML signature wrapping. Since SQL injection attacks have occurred since the advent of SQL databases, the defense mechanisms have evolved over time. Halfond et al. [9] categorized the early research into detection-focused and prevention-focused. Detection-focused techniques combine code checking with various schemes such as static analysis, dynamic debugging, blackbox testing, and taint-based analysis. Prevention-focused techniques prevent execution of any malicious SQL queries by means of a proxy filter and applying instruction-set randomization (ISR).

b) Device-side injections

Various methods for injecting malware into IoT devices exist since they have highly heterogeneous hardware and firmware. The most common approach to remotely injected malware is to exploit the zero-day vulnerabilities that can lead to remote code execution (RCE) or command injection.

Possible Solutions:

For IoT devices, the main threat of injection attacks consists of firmware modification attacks. As of today, there is only limited information available to defend against this attack. To the best of our knowledge, Cui et al. [10] was the first to propose the defense mechanisms to mitigate such attacks. Inspired by the idea of address space layout randomization (ASLR) and ISR, they proposed the autotomic binary structure randomization (ABSR) that takes arbitrary executables or firmware as the input and outputs a variant of the original with a reduction of unused codes to minimize the attack surface.

4. Authentication and authorization attacks

Authentication is the action of verifying the user identities of people who request certain services. Authorization is the process of determining the access rights and privileges of an entity, confirming that the entity behaves according to its rights without crossing boundaries.

Authorization is usually preceded by authentication for identity verification. In edge computing, authentication is generally performed between edge devices and edge servers.

If an attacker intends to directly access protected edge servers or edge devices, it will be blocked by the authentication system. Therefore, an attacker seeks methods to bypass the authentication process by performing an unauthorized access.

These types of attacks can be grouped into three major categories:

a) Dictionary attacks

These types of attacks target the Default and Weak Credentials by using a dictionary containing the most-used credentials/passwords. They are generally known as brute force attacks.

Most IoT devices have web-based management interfaces where users can log in to adjust settings, install updates, and perform other routine tasks. These interfaces often come with a default password that users are encouraged to change, but in reality, very few users actually do this, making IoT and edge computing devices easy targets for attackers.

Possible Solutions:

Expand password policy for enforcing strong passwords on IoT devices. Use of biometrics wherever possible, as it is much more difficult to breach. Two-factor authentication can also help immensely by adding another layer, which increases the cost of dictionary attacks, making them impractical.

b) Exploiting weaknesses in authentication protocols

Various methods for injecting malware into IoT devices exist since they have highly heterogeneous The previously mentioned dictionary attacks generally require high resource consumption and have a low success rate, hence attackers tend to work more efficiently by discovering the design flaws of the authentication protocols.

There have been studies that observed the weak binding vulnerability existing in the WPA enterprise authentication protocol, as explained in reference[11]. Even TLS and WPA2 protocols have been found to be prone to attacks.

Possible Solutions:

To defend against such attacks, either we can enhance the security of communication protocols or secure the cryptographic implementations.

Securing the communication

- One study [18] suggested using active jammer and wireless packet injection to inhibit the brute force attack, which decrypts the WPA traffic.
- Another approach is to revise the original key exchange process in the WPA/WPA2 protocol by adopting the public key cryptography. This can reduce the threat of several vulnerabilities, including the evil twin, as explained in reference [19].

Securing the cryptographic implementations

- A black box verification mechanism has been suggested to prevent possible hostname impersonations, as explained in reference [20].

To prevent TLS attacks, TLS implementations can be scanned using symbolic execution to detect if it is vulnerable to various well-known attacks like Logjam or the Triple Handshake.

c) Exploiting weaknesses in authorization protocols

These attacks exploit design weaknesses or logical flaws in authorization protocols in order to achieve unauthorized access to sensitive resources or perform privileged operations (also known as the overprivileged issue).

OAuth is commonly used for authorization in the industry and in edge computing as well. It involves three entities: a user, a service provider, and a relying party. It allows the service provider access to the user's resources (stored in the relying party) only after the user grants access rights to the service provider.

While OAuth 1.0 is known to be broken, OAuth 2.0 is being widely used. Although OAuth 2.0 is not vulnerable in theory, many studies have proved that its incorrect implementation can pose danger.

One study [12] identified that the OAuth protocol in 59.7% of mobile applications was incorrectly implemented.

Possible Solutions:

As mentioned before, OAuth1.0 is vulnerable and should be replaced with OAuth2.0.
Even with OAuth2.0, extra caution is advised while implementing. A static code analysis method can be used to check and fix the OAuth implementation vulnerabilities based on three OAuth service providers: Google, Facebook, and Sina (as explained in reference [13]).

Misuse of the current OAuth APIs can be prevented by using an application-based OAuth Manager framework, as explained in reference [14].

Conclusion

In this paper, we provided a comprehensive study of edge computing, its applications and most dangerous threats, as well as the corresponding defense mechanisms that can be applied in practice.

In this section, we would like to discuss the current state and the major challenges in securing an edge computing system that any practitioner should be aware of.

The primary goal of edge computing is to provide a more efficient and lightweight computing platform for emerging applications such as IoT and smart cities. This shifts the focus from security to performance when designing an edge computing application. Thus, we need to prioritize security when designing any edge computing infrastructure.

Due to the heterogeneity of edge devices with diverse OSes and software, different network topologies, and disparate communication protocols, it is possible that the security frameworks designed for one edge computing application may not be directly migratable and applicable to another scenario.

Existing defense mechanisms in the edge computing landscape are both isolated and passive. They are isolated because each defense mechanism discussed above may only be effective in countering one attack or a few, but less effective for the majority of attacks. They are passive because most of the defense solutions are executed based on predefined rules and lack the ability to conduct autonomous and active defense actions.

These two weaknesses result in a rigid and fixed defense surface, forcing most current defense solutions to adopt a philosophy of "detect then patch," which might be helpful only once attacks are detected.

The research and development on edge computing security are still in their initial stages. Driven by emerging applications and advances in modern cryptography, innovative designs and implementations to secure edge computing systems will see more growth in the foreseeable future.

About the Authors

Ashish Garg is a Manager, Engineering at GlobalLogic, and a security practitioner. He has extensive experience in digital transformation and Web Application development.

Bipin Gupta is a Consultant at GlobalLogic with an expertise in Web application development and Client handling.

References

[1] The Growth in Connected IoT Devices

[2] Wifi Network Security Statistics/Graph - Infographics by Graphs.net. Accessed on Dec. 7, 2016.

[3] NVD - cve-2010-3972 detail (2010)

[4] Z. L. Chua, S. Shen, P. Saxena, and Z. Liang, "Neural nets can learn function type signatures from binaries," in Proc. 26th USENIX Secur. Symp. Canada: USENIX Association, 2017, pp. 99–116.

[5] F. Zuo, X. Li, Z. Zhang, P. Young, L. Luo, and Q.Zeng, "Neural machine translation inspired binary code similarity comparison beyond function pairs," in Proc. NDSS, 2019, pp. 1–15.

[6] T. Frassetto, P. Jauernig, C. Liebchen, and A.-R. Sadeghi, "IMIX: In-process memory isolation xtension," in Proc. 27th USENIX Secur. Symp. (USENIX Security). Baltimore, MD, USA: USENIX Association, 2018, pp. 83–97.

[7] S. Shirali-Shahreza and Y. Ganjali, "Protecting home user devices with an SDN-based firewall," IEEE Trans. Consum. Electron., vol. 64, no. 1, pp. 92–100, Feb. 2018.

[8] S. S. Clark et al., "WattsUpDoc: Power side channels to non-intrusively discover untargeted malware on embedded medical devices," presented at the USENIX Workshop Health DC, USA: USENIX, 2013.

[9] W. G. J. Halfond, J. Viegas, and A. Orso, "A classification of SQL injection attacks and countermeasures," Tech. Rep., 2006.

[10] A. Cui, M. Costello, and S. Stolfo, "When firmware modifications attack: A case study of embedded exploitation," in Proc. NDSS, 2013, pp. 1–13.

[11] A. Cassola, W. K. Robertson, E. Kirda, and G. Noubir, "A practical, targeted, and stealthy attack against WPA enterprise authentication," in Proc. NDSS, 2013, pp. 1–15.

[12] E. Y. Chen, Y. Pei, S. Chen, Y. Tian, R. Kotcher, and P. Tague, "OAuth demystified for mobile application developers," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2014, pp. 892–903.

[13] R. Yang, W. C. Lau, and S. Shi, "Breaking and fixing mobile app authentication with OAuth2.0-based
protocols," in Proc. Int. Conf. Appl. Cryptogr. Netw. Secur. Springer, 2017, pp. 313–335.

[14] M. Shehab and F. Mohsen, "Securing OAuth Implementations in Smartphones," in Proc. 4th ACM CODASPY, USA, 2014, pp. 167–170.

GlobalLogic®

GlobalLogic, a Hitachi Group Company, is a leader in digital product engineering. We help our clients design and build innovative products, platforms, and digital experiences for the modern world. By integrating our strategic design, complex engineering, and vertical industry expertise with Hitachi's Operating Technology and Information Technology capabilities, we help our clients imagine what's possible and accelerate their transition into tomorrow's digital businesses. Headquartered in Silicon Valley, GlobalLogic operates design studios and engineering centers around the world, extending our deep expertise to customers in the automotive, communications, financial services, healthcare & life sciences, media and entertainment, manufacturing, semiconductor, and technology industries.



www.globallogic.com