AFRICAN SOCIAL AND EDUCATIONAL JOURNAL
FACULTY OF BUSINESS ADMINISTRATION
IMO STATE UNIVERSITY
NIGERIA
VOL. 8 NO. 1 MARCH 2019

EMBEDDED SYSTEMS SECURITY

BASAKY, FREDERICK DUNYA, PhD.
College of Information and Communication Technology
Salem University, Lokoja
Kogi State

Abstract

An Embedded system is an electronic product that contains a microprocessor (one or more) and software to perform some constituent functions within a large entity. Embedded systems security is a new and emerging area of research. It is the meeting point of many disciplines such as electronics, logic design, embedded systems, signal processing and cryptography. It is closely related to the area of information and software systems security because software is an integral component of any embedded system. Not long ago, it was thought that only software applications and general purpose digital systems i.e. computers were prone to various types of attacks against their security. The underlying hardware, hardware implementations of these software applications, embedded systems, and hardware devices were considered to be secure and out of reach of these attacks. However, during the previous few years, it has been demonstrated that novel attacks against the hardware and embedded systems can also be mounted. Not only viruses, but worms and Trojan horses have been developed for them, and they have also been demonstrated to be effective. Whereas a lot of research has already been done in the area of security of general purpose computers and software applications, hardware and embedded systems security is a relatively new and emerging area of research. This paper explains the details of knowing and understanding the security risks associated with embedded systems and how to mitigate them. It also provides details of various types of existing attacks against hardware devices and embedded systems, analyzes existing design methodologies for their vulnerability to new types of attacks, and describes solutions and countermeasures against them for the design and development of secure systems. The paper explains some of the most current security risks associated with embedded systems and how to address them in personal devices. It also looks at some of the security concerns in embedded systems around the world and how their architects are working to keep them secure

Introduction

The invention of microprocessors and microcontrollers has replaced the analog components and devices of most electronic equipment developed today with digital components for design implementation. Embedded systems are the propulsive force for technological development in many fields such as automotive, healthcare, and industrial control in the developing post-PC generation. Due to rapid increase usage of computational and

networked devices in all facets of our lives in a prevalent and invisible way, it is important the securities of these systems are dependable and reliable.

Review

With the growing popularity of embedded systems, concerns about security and privacy of these systems have risen dramatically in a short period of time. While operating systems like Windows and Mac OS have well-known antivirus solutions that are constantly updated with new virus definitions, the ubiquity and modularity of embedded systems make them more difficult to secure.

In a world that increasingly relies on embedded systems, security threats can result in serious damage in the best cases and utter chaos in the worst. In Poland, a 14-year-old high school student spent weeks trespassing at a train yard, gathering information that he used to convert a TV remote controller into a control device for the track points. The youth, now facing juvenile disciplinary hearings, caused 4 trains to derail and injured 12 in the process.

An embedded systems application may not carry the inherent risks of one used to operate a train yard, but it's still important to know and understand the security risks associated with embedded systems and how to mitigate them. This article explains some of the most current security risks associated with embedded systems and how to address them in personal devices. We'll also look at some of the security concerns in embedded systems around the world and how their architects are working to keep them secure.

Key Terms and Definitions

Covert channel: is a hidden communication channel between a malicious entity and a victim system which is conventionally not used for transfer of information.

Electrically Erasable Programmable Read-Only Memory (EEPROM): is a type of Read Only Memory (ROM) which is used in embedded systems and computers to store data or boot program which must not be destroyed when power is turned off. The data or program can only be erased or programmed electrically.

Embedded system: is a digital system which is designed to perform a specific task or set of tasks as compared to a general purpose computer which allows user to perform various types of tasks by installing new software on top of its operating system.

Field Programmable Gate Array (FPGA): is a type of integrated circuit which can be programmed by the design engineer to implement a particular logic at the hardware level. An HDL is used to program an FPGA.

Finite state machine (FSM): is an abstract mathematical and behavioral model of a system consisting of a number of finite states that the system may switch to and a finite number of inputs that cause the transition from one state to the other. An FSM is also commonly known as finite state automation or simply state machine.

Fuzz Test: or fuzzing is a testing technique in which the system is provided with various combinations of invalid input and behavior of the system is observed.

Hardware Description Language (HDL): is a computer programming language which is used to describe the logic and flow of an electronic circuit. Examples of HDL are VHDL and Verilog.

Hardware Trojan horse: is a bug or backdoor in the form of a tangible circuit or software piece of code in the control program of an electronic system. It may allow an unauthorized user to communicate with the system and control it.

S-Box: is an acronym for Substitution-Box and is a type of lookup table which is used in a symmetric key encryption algorithm to perform substitution operation on a given plain text. The box receives m number of input bits and according to some lookup function, translates them to an n number of output bits.

Side channel attack: is a type of attack on an embedded system which treats the embedded system as a black box and tries to infer hidden information by observing the behavior (timing, power consumed, etc.) and output of the system by feeding it various types of inputs.

Smart card: is a small pocket-sized plastic card which has an integrated circuit (usually an embedded processor) inside it.

Subscriber identification module (SIM): is a smart card used in GSM and CDMA based cellular telephones to allow users to switch phones by simply switching the SIM.

Discussion/ Embedded Systems Security

Due to increased usage of embedded systems in our daily lives, it is important that these systems and the components used in them are greatly trustworthy and secured. The security of embedded systems is an area of research that is now emerging.

Embedded system security is the reduction of vulnerabilities and protection against threats on software running on embedded devices. Like security in most IT fields, embedded system security involves a conscientious approach to hardware design and coding as well as added security software, an adherence to best practices and consultation with experts.

In the past, the large number of embedded operating systems and the fact that these systems did not typically have direct internet communication provided some degree of security, both through obscurity and the fact that they were not convenient targets.

Generally, many of the hardware and hardware systems controlled by embedded software have not been easily interfaced with as they had little need to be exposed. Trends like machine to machine (M2M) communication, the Internet of things (IoT) and remotely controlled industrial systems, however, have increased the number of connected devices and simultaneously made these devices targets. As attacks on embedded system have become common, it has become increasingly important to protect them.

In 1971, the first microprocessor Intel 4004 was invented and other invention in this field emanated to the development of computer systems and embedded devices. Computer systems and embedded devices have software as their essential component. Generally, computer has software called operating system which acts as an intermediary between programs (software) and the computer hardware.

Computer programs were subjected to various types of threats and attacks. The Creeper virus developed by Bob Thomas in 1971 was the first attack on software and has continued till date. Most of the methods and approaches used in attacking software application is also application to embedded devices, particularly, in the firmware component. An embedded system can be attacked by injecting malware at some point. Once installed, this can collect confidential data, change a system's behaviour or induce unpredictable actions.

The Biggest Security Threats Facing Embedded Designers

Engineers building embedded system are dealt with a number of threats to the applications that they build for the internet of things (IoT). The accessibility of IoT subsystems such as commercial networked HVAC systems, wireless base stations (e.g., small cells), implanted medical devices and their controllers, smart automobiles and the emerging networked transportation infrastructure, home and industrial-infrastructure network gateway systems, and remote industrial sensors by hackers is One of the major threats to the embedded systems. There are cases of attacks by end users on an IoT subsystem they own or have access to, like their smart residential electric meter, with the desire to have a reduced electric bill; or their connected car, to bypass emissions controls. Some attackers may be propelled economically, or could be nation states preparing for cyber war.

Factors that make IoT endpoints especially vulnerable to security threats include:

***Networked**:* IoT endpoints may be remotely accessible from nearly anywhere in the world via the internet or other (e.g., phone) networks. Wireless connections, used in many IoT devices, are especially vulnerable.

***Fielded**:* IoT devices are often physically accessible, as well as interconnected. This exposes them to additional hardware attacks that do not usually need to be considered for systems that may be networked but are physically protected by "guns, guards, and gates."

***Available**:* Samples are often easily available through purchase or theft that can be analyzed at the attacker's leisure.

Combining Hardware and Software to Secure the IoT

Software security, alone, has proven relatively unsatisfactory in protecting networked devices against known and freshly discovered threats (so-called "zero day" vulnerabilities), and is totally inadequate for the additional threats posed to fielded systems. What is needed is a combination of software *and hardware security. For example, today's SoC FPGAs can be used to implement a hardware security scheme that complements the software and strengthens the system. Ideally, the hardware and software solution should combat three types of security: design security, hardware security, and data security.*

***Design security**:* This includes IP protection and ensuring that configuration bit streams and firmware are encrypted and protected. Designs need to incorporate a method to ensure that overbuilding or cloning of the design is not possible. Field updates to processor firmware or FPGA configurations need to be authenticated and the payload kept confidential.

***Hardware security**:* Designers also need to certify that user-accessible devices are resistant to physical attacks. For example, differential power analysis (DPA) attacks can extract keys and other vital device information. System boot-up needs to be kept secure, not just from remote network-based attacks, but also where the adversary has physical access.

***Data security**:* This element ensures that communications into and out of the system are authentic and secure, and sensitive data stored in the system cannot easily be extracted.

Embedded-system program managers and development teams must design these types of protections into their products while best leveraging the characteristics of the underlying platform. The result should be a robust protection network with no single point of failure. Key methods for achieving this goal include:

***Risk assessment**:* Perform a detailed system security evaluation early in the architecture design phase, to assess critical system data/functions, discover vulnerabilities, enumerate threats, and outline the likelihood and consequence of system compromise.

***Protection planning**:* Using risk assessments and any other compiled data, developers should seek to understand protection implementation costs and design options for mitigating identified system vulnerabilities and ensuring successful system verification and validation.

***Attack scenario testing**:* This can include a black-box approach, pitting experienced reverse engineers with state-of-the-art attack tools against a system in a deployed setting to reveal vulnerabilities that cannot otherwise be found during most other evaluation exercises.

***Side-channel analysis and mitigation**:* Side-channel attacks like differential power analysis (DPA) are currently the most practical method for compromising cryptography implementations. It is important to perform measurable, objective, and repeatable testing for resistance to side-channel attacks for applications where adversaries have the ability to observe side channels (i.e., power draw, timing, EM emanations) during on-device cryptographic operations.

Typical Threats to Embedded Systems

The following are common attacks on embedded systems that should be considered during the architecture and design implementation engineering phases.

Network attacks: Though fielded systems are subject to new threats, all the existing battery of network attacks still apply, too. Ideally, all network communication is authenticated and encrypted using well-established protocols such as TLS. A public key infrastructure (PKI) can be used by both remote endpoint devices (clients) and servers to ensure that only communications from properly enrolled systems are accepted by the parties to the communication. A strong hardware root of trust can provide this secure "identity" for the system, providing unique-per-device keys linked to the hardware and certified in the user's PKI.

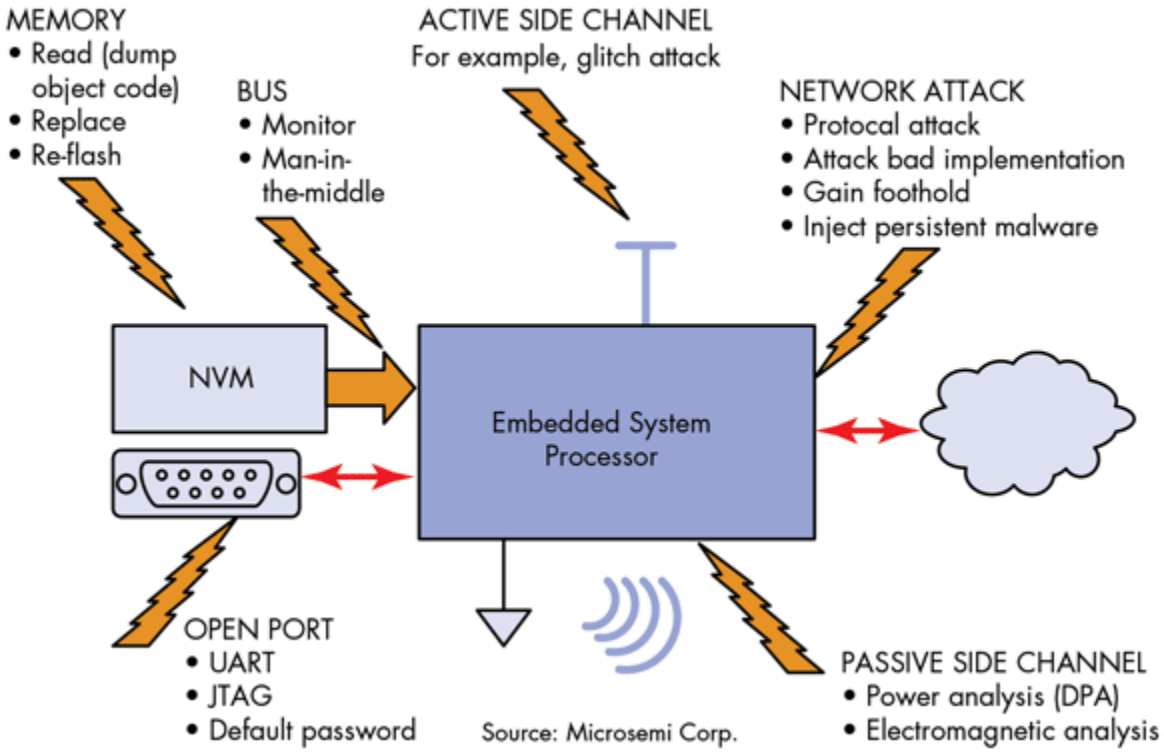
***Active side-channel attacks**:* The most common active attack is to use voltage glitches on the power supply of the processor to cause an "interesting" malfunction at a critical juncture in the execution of the embedded program. For example, the infamous attack on the Sony Playstation, which started a chain of events that allegedly cost Sony $1 billion before it all played out, started as a glitch attack that allowed the hacker to enter a privileged processor state where he could then dump all the code, whereupon a cryptographic implementation flaw was found. Game over!

Memory and bus attacks: If the hardware is physically available and insufficiently protected, it may be possible just to read the contents of memory directly from an external programmable read-only memory (PROM) or external RAM memory chip, or by probing the connecting bus. It is generally good practice, and not that difficult, to encrypt and authenticate all static data such as firmware stored in PROMs. Another memory attack is called Cold Boot Attack where the memory (a bank of SDRAM chips, for example), is chilled, quickly removed, and read on another system controlled by the attacker. The cold chips hold remnants of the data even during the short interval where they are unpowered. Thus, it is best not to store critical secrets such as cryptographic keys in off-chip memory. In cases where higher levels of security are justified, external volatile memory can be encrypted.

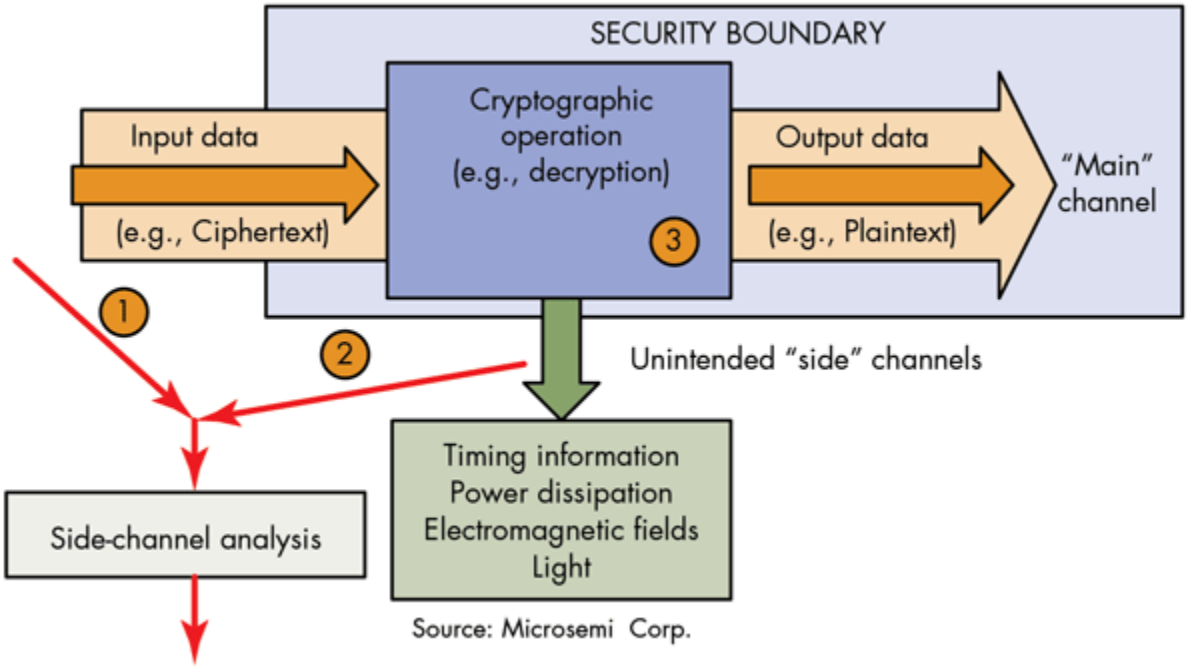
Some memory attacks can be performed remotely, such as with the Heartbleed bug introduced in a 2011 OpenSSL update that at its peak allowed remote reading of RAM working

memory in an estimated half million secure web servers worldwide. The Heartbleed bug was discovered and announced publicly by Google and independently by Codenomicon in April 2014. Technologies such as whitebox cryptography can be brought to bear on this type of memory attack.

Reflashing, or changing the contents of memory, either remotely or with physical access can, in what is called a "persistent" attack, bypass any software-based security. Since the malware is installed in non-volatile boot-up memory, even resetting the system does not clear away the malware. It is important to boot the embedded processor securely by verifying the authenticity of all code in rewritable memory, or A secure FPGA can protect debug and other factory test ports from hackers at the hardware level while still providing access to authorized personnel. The access keys can be secured against extraction by the FPGA, providing much better protection than software alone can.



2.3.5 ***Passive side-channel attack:*** Simple and Differential Power Analysis (SPA and DPA), and their electromagnetic equivalents (SEMA and DEMA) can extract secret keys used inside a processor or FPGA, if suitable countermeasures are not implemented



Security Requirement for Embedded Systems/Devices

Various attack on embedded systems/devices lately recorded ranges from hacked vehicle, anti-theft and control systems to hijack printers that sent copies of document to the hacker's computer. Password protected logins and encrypted protocols for example SSH or SSL included in embedded systems/devices is not enough to secure them.

Large enterprises use multiple layers of protection as the basis of their security which is proven and well established security principles. These layers include firewalls, authentication/encryption, security protocols and intrusion detection/intrusion prevention systems. With all these, embedded systems do not have firewall as it used to be assumed that embedded systems are not vulnerable to attacks as its reliability is on password authentication and security protocols. Presently the assumptions have been negated as the attacks on embedded systems is on a high rate, thus the need for greater security measures.

For the past two decades, large enterprises have majorly focused on cyber security; engineers building embedded systems just recently switch focus to cyber security. The best option for engineers building embedded computing devices/systems is to learn from enterprise cyber security experience or from their playbook.

A security solution for embedded devices must ensure the device firmware has not been tampered with, secure the data stored by the device, secure communication and protect the device from cyber-attacks. This can only be achieved by including security in the early stages of design.

There is no one one-size fits all security solution for embedded device. Security requirements must take into consideration the cost of a security failure (economic, environmental, social, etc.), the risk of attack, available attack vectors, and the cost of implementing a security solution. Features that need to be considered are:

Security Features and Implementations

Secure Boot: Achieved using cryptographically signed code from the manufacturer along with hardware support to verify code is authenticated. This ensures that the firmware has not been tampered with.

Secure Code update: ensures that the code on the device can be updated for bug fixes, security patches, etc. Use of signed code (secure boot) ensures that malicious code cannot be introduced into the system.

Data security: Prevent unauthorized access to the device, encrypted data storage and/or encrypted communication.

Authentication: All communication with the device should be authenticated using strong passwords (at minimum) or use of an authentication protocol such as Kerberos.

Secure Communication: Communication to or from the device needs to be secured using encrypted communication (SSH, SSL etc.). Care must be taken to avoid the use of insecure encryption algorithm. i.e., 40 bit encryption keys that were once state-of-the-art are no longer considered secured.

Protection against cyber-attacks: Embedded firewalls provide a critical layer of protection against attacks. A firewall can limit communication to only known, trusted hosts, blocking hackers before they can even launch an attack. A layer of defence to protect against common attacks such as packet flood attacks, buffer overflow attacks and known protocol exploits. A firewall can implement many of these protections, but some must be built into the embedded applications.

Intrusion detection and security monitoring: Existing embedded devices can be attacked and no one would even know they are being attacked. A hacker could execute thousands or millions of invalid login attempts without any knowledge of an attack having taken place. Embedded devices must detect and report invalid login attempts and other potentially malicious activities. Note: monitoring requirements for embedded devices are very different than enterprise requirements. The IDS requirements for embedded devices will depend upon the protocol supported by the device.

Embedded security management: Integration with a security management system allows security policies to be updated to mitigate against known threats.

Device tampering detection: Some new processor/board designs include device tamper detection capabilities. They provide the ability to detect that the seal on the device enclosure has been broken, indicate that someone may be attempting to tamper with the device.

Integrating Security into the Device

Building protection into the device itself provides a critical security layer - the devices are no longer depending on the corporate firewall as their sole layer of security. In addition, the security can be customized to the needs of the device.

Security needs to be considered early in the design of a new device or system. Support for secure boot or device tamper detection requires specific hardware capabilities. Since hardware is typically selected early in the design phase, this capability must be considered very early in the process. Since many embedded devices are deployed outside of the standard enterprise security perimeter, it is critical that security be included in the device itself.

Many of today's modern embedded devices and systems are complex connected computers charged with performing critical functions. Including security in these devices is a critical design task. Security features must be considered early in the design process to ensure the device is protected from the advanced cyber-threats they will be facing.

Embedded Security Challenges

Embedded devices are very different from standard PCs. They are fixed function devices designed specifically to perform a specialized task. Many of them are designed using a specialized operating system such as VxWorks, MQX or Integrity, or a stripped down version of Linux. Installing new software on the system in the field either requires a specialized upgrade process or is simply not supported. In most cases, these devices are optimized to minimize processing cycles and memory usage and do not have a lot of extra processing resources available.

As a result, standard PC security solutions won't solve the challenges of embedded devices. In fact, given the specialized nature of embedded systems, PC security solutions won't even run on most embedded devices.

Other Challenges for Embedded Device Security include:

Critical functionality: Embedded devices control transportation infrastructure, the utility grids, communication systems and many other capabilities modern society relies upon. Interruption of these capabilities by a cyber-attack could have catastrophic consequences.

Replication: Once designed and built, embedded devices are mass produced. There may be thousands to millions of identical devices. If a hacker is able to build a successful attack against one of these devices, the attack can be replicated across all devices.

Security assumptions: Many embedded engineers have long assumed that embedded devices are not targets for hackers. These assumptions are based on outdated assumptions including the belief in security by obscurity. As a result, security is often not considered a critical priority for embedded designs. Today's embedded design projects are often including security for the first time and do not have experience and previous security projects to build upon.

Not easily patched: Most embedded devices are not easily upgraded. Once they are deployed, they will run the software that was installed at the factory. Any remote software update capability needs to be designed into the device to allow security updates. The specialized operating systems used to build embedded devices may not have automated capabilities that allow easy updates of the device firmware to ensure security capabilities are frequently updated.

Long life cycle: The life cycle for embedded devices is typically much longer than for PCs or consumer devices. Devices may be in the field for 15 or even 20 years. Building a device today that will stand up to the security requirements of the next two decades is a tremendous challenge.

Proprietary/industry specific protocols: Embedded devices use specialized protocols that are not recognized and protected by enterprise security tools. Enterprise firewalls and intrusion detection system are designed to protect against enterprise specific threats, not attacks against industrial protocols.

Deployed outside of enterprise security perimeter: Many embedded devices are mobile or are deployed in the field. As a result, these devices may be directly connected to the Internet with none of the protections found in a corporate environment.

Techniques to Protect Embedded Systems from Hacking

Cyber-attacks on embedded systems are on the rise. Embedded systems control numerous commonly used devices, including tablets, medical devices, industrial instrumentation, automobiles, and more. For embedded system security, you can use a number of techniques to reduce vulnerabilities and provide protection against threats against embedded devices.

Believing that embedded systems fell below the radar of serious hackers led to waves of attacks. Hackers can exploit these systems to steal intellectual property, copy designs, gain access to proprietary information about companies and their customers, utilize them as platforms to propagate further attacks, and even cause real world physical damage and harm to humans. Utilizing proper techniques, you can make your embedded system more secure and fight back against aggressive hack attempts.

Anti-Hacking Techniques

The following are a number of embedded system security techniques in use today. With upfront planning in the requirements definition phase and then following through with implementation during product design, embedded systems can be made far more secure.

- Utilize a tamper-resistant enclosure around the design. Physical and electronic features, such as deadman switches or anti-tamper meshes, can be added to the enclosure that detect opening of the enclosure, or perhaps other intrusions such as drilling. These features can connect to circuitry that recognizes and reacts to the tamper. Board hardware then can react with countermeasures, such as resetting the system or erase passwords, codes, and other critical information.
- Implement circuitry distributed around the circuit board whose operation changes or responds to tamper. For example, a microcontroller placed on the circuit board can connect to various signals that are routed strategically around critical areas. If a hacker tries to access the signals by cutting the board, the operation of the signal changes (it likely is cut off and stops operating), triggering a tamper detection event and reaction.
- Utilize secure boot features of the microprocessor to authenticate firmware prior to execution. Prior to the start of boot image execution, the processor verifies that the boot image has been signed with a predetermined cryptographic key that is stored within the processor or other secure storage mechanism. An image without the proper key signature is considered improper and is not executed.
- Enable a Trusted Execution Environment (TEE) within the embedded system's microprocessor. For example, ARM processors utilize Trust Zone technology, where the processor and its peripheral set can be divided into secure and non-secure regions. The secure region utilizes encryption, authentication, security keys, and digital rights management features. The combination of secure boot and Trust Zone implements a root and chain of trust that is resistant to rogue images being loaded and executed.

- Employ cryptography to encrypt and secure data being communicated. For example, strong encryption methods such as WPA-2 can be used for Wi-Fi transmission of data.
- Store data and other critical information in secure storage memories. This may be accomplished be encrypting data within the microprocessor or other encryption engine prior to writing to the memory (and decrypting the data after being read). Where possible, data may be stored within the processor itself, minimizing data transmission around the circuit board and avoiding probe attacks.
- Hardware accelerated cryptography engines perform computationally intensive cryptographic calculations far more efficiently than general purpose CPUs. This enhances cryptographic security be allowing more calculations per second and enables longer key generation. Many microprocessors include hardware cryptographic engines. For example, the NXP i.MX 6 enables SHA-256 hashing and AES keys up to 256 bits.
- Hardware random number generators can ensure strong and truly random data encryption keys and protect against protocol replay. Random numbers generated from software algorithms, while meeting statistical requirements for randomness, can be reconstructed once the algorithm is known and allow for a valid data transmission to be replayed, or repeated. Most current generation microprocessors have hardware random number generators.
- Secure real-time clocks provide reliable tamper-protected time sources. This is critical for applications that require accurate time stamps, such as billing for utilities, sensor data collection, point of sale terminals, medical devices, and more. Secure real time clocks are provided within some microprocessors and within standalone integrated circuits, and guard against unauthorized time changes, whether from malicious code or ESD/noise events.
- Secure debugging features protect an embedded system from attack via common debug mechanisms such as JTAG or Android Debug Bridge (ADB). The hardware being debugged must be capable of enabling debug port access via a unique and predetermined code or key. Leaving a debug port unprotected allows a hacker to observe program execution, read registers or memory locations, and even insert hostile code or change register/memory contents.
- For embedded system security, be careful when using the network for loading fixes to bugs, as well as remote updates. Hackers often use this to install code that gives them visibility to traffic going into and out of the system. The problem is that once the code is installed at the heart of the system, detection and removal are nearly impossible. The reason is that the code now controls commands for updates and detection. Controlling the update mechanism through a system such as InHand Device Manager provides the users total control of the update process.

Optimally, embedded system security must be planned at the requirements phase of any product design. This is required for the intentional selection of a processor with the appropriate security features and to implement hardware level detection. In some cases, it may be possible to update embedded system software to enable and utilize existing features that enhance security, such as encryption and authentication engines.

InHand Electronics specializes in the design of embedded systems and single board computers that are deployed in military, medical, and industrial applications, all areas where

security is imperative for health, safety, and confidentiality. Our Security Smart suite of security technologies enables ground-up protection of embedded systems from hackers.

New Knowledge: Future Research Directions

As embedded systems security is an emerging area of research, the hardware security research community is expected to discover and develop new forms of attacks on the hardware devices and embedded systems. This will further fuel the research on the countermeasures and protection schemes against these attacks. As a result, there may be a paradigm shift in the design of hardware devices and embedded systems particularly those used in defence applications. For example, design and implementation of an IC may go through fundamental changes from an abstract level description to semiconductor level fabrication so as to incorporate new security measures. For this to happen, the research community needs to come up with reliable and robust techniques which can be implemented at every layer of abstraction.

Embedded systems security shares a few common traits with software security. Therefore, in these common areas existing security techniques and methods may be applied. However, a few of the traits of embedded systems security are different from those of software security. Therefore, new algorithms and techniques will need to be developed for these areas. For example, research community will need to build secure operating systems which can be deployed in embedded systems which are usually resource constrained. This will include implementing security in all the critical functions performed by an operating system.

In many instances, attacks on embedded devices are possible only if physical security of the device is compromised. For example, in the case of side channel attacks on smart card, the attacker first needs to get a copy of the smart card which she intends to attack. Therefore, new security techniques will need to be developed which can prevent physical tampering of the device and ensure that the information stored in it remains secure. Embedded devices are usually limiting in resources and most of the existing cryptographic algorithms are computation intensive. Implementation of security with the heavy algorithms results in performance degradation. Therefore, lightweight cryptographic algorithms and protocols are needed which are tailored to run on embedded devices with limited resources.

Conclusion

Embedded systems are finding widespread uses in every sphere of our lives and their security has become an important research issue. In this chapter, we have discussed the background and current state of research on the threats and attacks being developed against embedded systems. The hardware attacks can be mounted at any of the layers of abstraction involved in the fabrication of the device with varying degrees of success. We have also discussed various countermeasures against these attacks.

Recommendation

If you've been left with more questions than answers when it comes to security for your devices, you're not alone. Connected devices and the Internet of Things are expanding opportunities for hackers faster than they can be addressed, and keeping up with current threats is a constant struggle. Thankfully, we've saved our two most important pieces of advice for the end of this article so you can take them with you everywhere.

First, make sure you consider security throughout product development. Don't plan to implement security later, as this almost never happens, and you'll leave yourself open to the most basic attacks from "clever outsiders".

Second, ensure that the cost of penetrating your security system is greater than the benefit. If organized criminals and professional hackers can't profit from breaching your security features, it's much less likely that your security system will come under attack. Added security measures, especially those at the software level, like data encryption, increase the barrier to entry for attacks against your devices.

References

Alkabani, Y., & Koushanfar, F. (2008, July). Extended Abstract: Designer's Hardware Trojan Horse. In Proceedings of IEEE International Workshop on Hardware-Oriented Security and Trust, HOST 2008, pp. 82-83, Washington DC, IEEE Computer Society.

Anderson, R., (2001). Security Engineering: A Guide to Building Dependable Distributed Systems. England, John Wiley & Sons.

Biham, E., & Shamir, A. (1997). Differential Fault Analysis of Secret Key Cryptosystems. In Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology, Vol. 1294, pp. 513-525, Springer-Verlag, London, UK.

Boneh, D., DeMillo, R. A., & Lipton, R. J. (1997). On the Importance of Checking Cryptographic Protocols for Faults. In Proceedings of the 16th Annual International Conference on Theory and Application of Cryptographic Techniques, pp. 37-51, Berlin, Springer-Verlag.

Debbabi, M., Saleh, M., Talhi, C., & Zhioua, S. (2006). Embedded Java Security: Security for Mobile Devices. Springer.

Gebotys, C. H. (2009). Security in Embedded Devices (Embedded Systems). Springer.

Hailes, S., Seleznyov, A., (2007). Security in Networked Embedded Systems. Springer.

Han, Y., Zou, X., Liu, Z., & Chen, Y. (2008). Efficient DPA Attacks on AES Hardware Implementations. International Journal of Communications, Network and System Sciences, 1: 1-103.

Koc K. C., & Paar, C. (1999). Proceedings of Cryptographic Hardware and Embedded Systems: First International Workshop, CHES'99 Worcester, MA, USA, Springer-Verl.

Kocher, P. (1996). Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems. In: N. Koblitz (Ed.), Proceedings of Annual International Conference on Advances in Cryptology, CRYPTO'96, vol. 1109 of LNCS, pp. 104–113, Springer-Verlag.

Kocher, P., Jaffe, J. & Jun, B. (1999). Differential Power Analysis. In Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO 99, Vol. 1666, pp. 388–397, Heidelberg, Germany, Springer-Verlag.

Lemke, K., Paar, C., & Wolf, M. (2010). Embedded Security in Cars: Securing Current and Future Automotive IT Applications. Springer.

Lessner, D., (2009). Network Security for Embedded Systems: A feasibility study of crypto algorithms on embedded platforms. LAP Lambert Academic Publishing.

Mangard, S., Oswald, E., & Popp, T. (2007). Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security). Springer.

Messerges, T. S., Dabbish, E. A. & Sloan, R. H. (2002, May). Examining Smart-Card Security under the Threat of Power Analysis Attacks. IEEE Transactions on Computers, Vol. 51, No. 5, pp. 541-552, IEEE Computer Society.

Mulliner, C., & Miller, C. (2009, July). Fuzzing the Phone in Your Phone. Black Hat, Las Vegas, NV. Mustard, S. (2006, January). Security of Distributed Control Systems: The concern increases. IEEE Computing and Control Engineering Journal, Vol. 16, Issue 6, pp. 19-25, UK, IET. Nedjah, N., & Mourelle, D. (Ed.). (2004). Embedded Cryptographic Hardware: Methodologies & Architectures. Nova Science Publishers.

Nedjah, N., & Mourelle, D. (Ed.). (2006). Embedded Cryptographic Hardware: Design & Security. Nova Science Publishers.

Parameswaran, R. G. R. S. (2008). Microarchitectural Support for Security and Reliability: An Embedded Systems Perspective. VDM Verlag.

Potkonjak, M., Nahapetian, A., Nelson, M., & Massey, T. (2009). Hardware trojan horse detection using gate-level characterization. In Proceedings of the 46th Annual ACM IEEE Design Automation Conference, CA, ACM.

Prevelakis, V., & Spinellis, D. (2007, July). The Athens affair. IEEE Spectrum, 44(7), 26-33.

Ray, J., & Koopman, P. (2009, July). Data Management Mechanisms for Embedded System Gateways. In Proceedings of IEEE International Conference on Dependable Systems and Networks, DSN'09, pp. 175-184.

Schmeh, K. (2003). Cryptography and public key infrastructure on the internet. West Sussex, England: John Wiley & Sons.

Standaert, F. X., Ors, S. B., Quisquater, J. J., & Prencel, B. (2004). Power analysis attacks against FPGA implementations of the DES. In Proceedings of the International Conference on Field-Programmable Logic and its Applications (FPL), LNCS 3203, (pp. 84-94). Heidelberg, Germany: Springer-Verlag.

Stapko, T., (2007). Practical Embedded Security: Building Secure Resource-Constrained Systems (Embedded Technology). England, Newnes.

Takahashi, J., & Fukunaga, T. (2010, January). Differential fault analysis on AES with 192 and 256-bit keys. In Proceedings of Symposium on Cryptography and Information Security. SCIS, Japan, IACR e-print archive.

Tiri, K., & Verbauwhede, I. (2004). A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation. In Proceedings of the Conference on Design, Automation and Test. IEEE Computer Society.

U.S.-Canada Power System Outage Task Force, (2004, April). Final Report on the August 14th, 2003 Blackout in the United States and Canada: Causes and Recommendations. US-Canada Power System Outage Task Force, The North-American Electricity Reliability Council, USA.

Van Eck, W., (1985). Electromagnetic Radiation from Video Display Units: An Eavesdropping Risk. Computers and Security, Vol. 4. No. 4, pp. 269-286, Oxford, UK, Elsevier Advanced Technology Publications.

Van Eck, W., (1985). Electromagnetic radiation from video display units: An eavesdropping risk. Computers and Security, 4(4), 269-286. Oxford, UK: Elsevier Advanced Technology Publications.

Verbauwhede, I. M. R. (Ed.). (2010). Secure Integrated Circuits and Systems. Springer.

Wright, P. (1987). Spycatcher – The candid autobiography of a senior intelligence officer. Australia: William Heinemann.

Zurawski, R. (2006, July). Embedded Systems Handbook. Taylor and Francis Group LLC.

https://internetofthingsagenda.techtarget.com/definition/embedded-system-security by Margaret Rouse.

http://www.newelectronics.co.uk/electronics-technology/how-to-protect-embedded-software-against-attacks/59422/ by Christophe Tremlethttps://www.tripwire.com/state-of-security/security-awareness/five-security-tips-to-protect-embedded-devices/ by CRAIG YOUNG

HTTPS://WWW.ICONLABS.COM/PROD/SECURITY-REQUIREMENTS-EMBEDDED-DEVICES-%E2%80%93-WHAT-REALLY-NEEDED.

HTTPS://WWW.INHAND.COM/PROTECT-EMBEDDED-SYSTEMS-FROM-HACKING/ BLOG BY Paul Willis.

https://www.electronicdesign.com/iot/biggest-security-threats-facing-embedded-designers article by Richard Newell, Senior Principal Product Architect, SoC Group, Microsemi Corp.