A vital analysis on Integrating Security in Embedded Systems

Vivek Purohit
Electronics and communication
B Tech Final Year
Shrinathji Institute of Technology and Engineering
Nathdwara, Rajsamand
vivekpurohit@hotmail.com

Garima Kothari
Electronics and communication
Assistant Professor
Shrinathji Institute of Technology and Engineering
Nathdwara, Rajsamand
kothari.garima@gmail.com

***Abstract*—In this new era, incorporating security in embedded system had become a major concern. Therefore, there is a need to design and develop such techniques, so that it can countermeasure for attacks. In this paper various security requirements, attack techniques and countermeasures for such attacks have been surveyed and reviewed.**

Index Terms—**embedded systems, security, virus, software, attacks.**

Station Switching.

I. INTRODUCTION

In this modern world, use of embedded systems in various applications has implacable importance of secure and fault tolerant system. Therefore, it is necessary for embedded devices to secure delicate information, ensuring availability and providing secure communication system. Reliability is directly related with security in embedded systems thus a system which is not secured is also an unreliable system. Basically security emerged as an issue related to networks and cryptography and embedded system designers consider it as an additional feature. Growing number of security laws has dictated that compromise on security can lead to disastrous consequences. Embedded systems are a mixture of Software and Hardware. So, in order to make a secure and reliable system, it is necessary that both components are made secure.
.

II. SECURITY REQUIREMENTS IN EMBEDDED SYSTEMS

The main feature of embedded system for which it is mainly deployed for is to access and process sensitive data and to provide vital functionality. For instance, let us assume that an embedded system has been implanted in a heart patient to monitor heartbeat, blood pressure and sugar level of that person. When it finds any deviation in blood pressure or sugar level, it transmits an alarm signal to doctor and when it found that heart beat is beyond a certain limit, it generates a mild shock to save life of that person. In this example, it is very obvious that system should be available twenty four hours a day and seven days a week. It is also important that when communicating with doctor, it should ensure privacy of patient. In general, an embedded system must ensure dependability, confidentiality, integrity and availability to consider as secure system. The basic security requirements of an embedded system are as follows:

Confidentiality- In order to ensure that sensitive information is protected against intended or accidental disclosure.
Integrity- In order to ensure that sensitive information is protected against intended or accidental corruption of data.
Availability-To make the device available to protect against intended or accidental actions that can cause automated information data to be unavailable to user when needed.
Dependability- Embedded systems often reside in Machines deploying embedded systems are expected to run continuously for years without any errors/faults and in some cases they are also require to recover by themselves if there is an error. Thus an embedded system should be dependable. Dependability is combination of:

- **Fault-Avoidance-**Developing a mechanism to avoid fault situation to occur. This is completed in designing
  phase.
- **Fault-Tolerance-** System should be required to work in normal condition even when a fault situation arises. This is completed through redundancy.
- **Fault-Removal-** During verification phase, it is ensured that system is error free.

III. CHALLENGES IN SECURE SOFTWARE DEVELOPMENT

There are many factors which can influence the sensitivity and availability of system administrative controls, physical barriers, being few of them, Whereas integrity of the computer system depends on the degree to which vulnerabilities have been eliminated from the system. With the requirements, discussed earlier, embedded system are also required to face threats like denial of service attacks, system tempering etc. which becomes more complex in presence of advanced techniques for breaking security, such as power analysis and fault analysis. Although Software solutions are sufficient to meet with with the computational demands of security processing in embedded system but they are more prone to security vulnerabilities. There are three main factors which make development of secure software a challenge i.e. Complexity, Extensibility and Connectivity. Let us examine each of these issues in brief. **Complexity-** Although Basics remain same but technology changes every two days, With this, software's are becoming more and more advanced. With each new line of code, new bugs and security risks are introduced in software. The complexity exacarbate when

unsafe programming languages (e.g., C or C++) are used, which are not able to protect against simple kind of attacks, such as buffer overflows.

Extensibility- Even platforms developed like .Net and Java are mainly to provide extensibility. Embedded system now can get updates or extensions from internet for example new cell phones by Sony Ericsson accept BIOS update through internet. Unfortunately, this nature of extensible systems makes it hard for software to prevent vulnerabilities from slipping in as an unwanted extension.

Connectivity- Nowadays modern embedded systems are coming with built in connectivity with internet. Because of this internet connectivity This internet connectivity a small problem can propagate and result in substantial security breaches, which also means that any attacker does not require a physical access to embedded system to launch attack and exploit vulnerable software.

IV. ATTACKS ON EMBEDDED SYSTEMS

Basically attacks on embedded systems can be classified in three categories i.e. software attacks, physical attacks and side channel attacks. Software attacks having largest share in total number of attacks on embedded systems and thus it are very difficult to protect against such attacks. Here we will discuss on software attacks and countermeasure against these attacks.

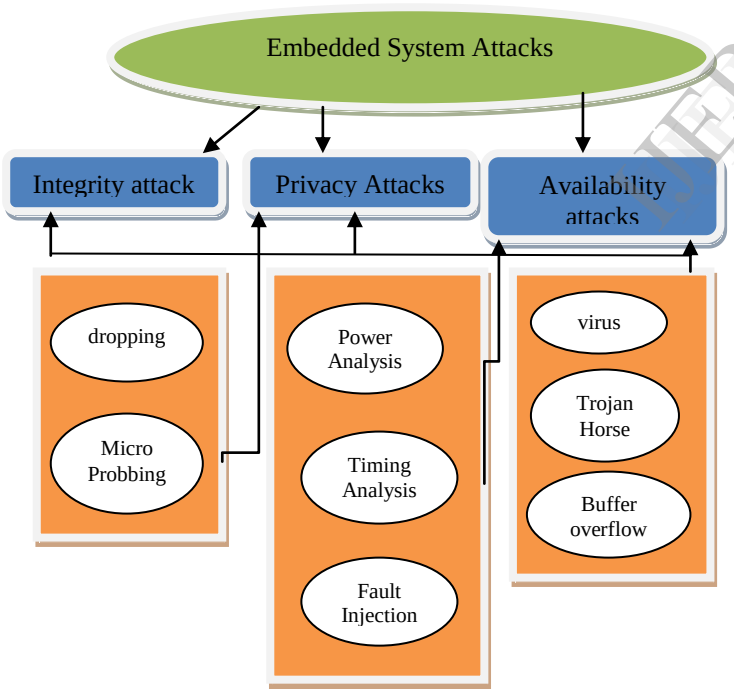

Figure 1 Embedded System attacks

Physical Attacks- Embedded system are divided in two categories:

1. System on circuit board
2. System-on-chip

On circuit board embedded systems, attacks can be launched by probe to eavesdrop on inter-component communications. Whereas when launching attack on system-on-chip, micro-probing techniques are required. Physical attacks are relatively difficult because they require expensive infrastructure and very complex techniques are used. Microcontroller is the most important part of any embedded system as it controls all the operation of embedded system. The attacks on the microcontroller can be possible via JTAG, it is necessary to disable access to the microcontroller's internals via JTAG before fielding the finished product

Side Channel Attacks- Side channel attacks depends on observing system properties e.g. time, power consumption while system is performing computations e.g. cryptographic operations. In certain embedded systems timing information can lead secret key, though timing information can give very little information but it has found that with proper study of timing sequence entire secret key can be found. Besides this power consumption can also lead to the entire secret key, well equipped labs have the equipment that can measure the changes in the power consumption with about 1% accuracy and are very less expensive. To avoid timing attacks one can add random timing delays to various operations, similarly we can overcome power consumption attacks by adding random noise or by proper shielding of the equipment but it can lead to increased cost of the equipment.

Software Attacks- Software attacks are the most common attacks in embedded systems. When compared with physical and side channel attacks, software attacks are very cheap and does not requires any big infrastructures thus making it an challenge for embedded system design. These attacks could further divided in three categories 1. Virus attacks 2. Buffer Overflow and 3. Exploiting Software Vulnerabilities.

1. **Virus, Worm and Trojan-** These attacks are executed through malicious agents like Virus, worm, Trojan.
2. **Vulnerability Exploitation-** A vulnerability allows the attacker to gain direct access to the end-system, while an exposure is an entry point that an attacker may indirectly exploit to gain access. The latest list of vulnerability as published by CERT is as follows:

Table 1: List of vulnerabilities as published by CERT

Vulnerability ID	Description
VU#298521	Sonic Wall Net Extender NELaunchCtrl ActiveX control stack buffer overflow
VU#446897	CUPS buffer overflow vulnerability
VU#180345	Microsoft Kodak Image Viewer code execution Vulnerability
VU#342793	RSA Keon cross-site scripting vulnerabilities
VU#871673	RealPlayer playlist name stack buffer overflow
VU#559977	Mozilla products vulnerable to memory corruption in the browser engine
VU#755513	Mozilla products vulnerable to memory corruption in the JavaScript engine

3. **Buffer Overflow-** Buffer overflow is the inability of the buffer to store the information, which results in adding more information to a buffer than it was designed to hold. An intruder may exploit this vulnerability to take over a system. This situation arises when buffer is used with poor boundary checks. Buffer bounds may be violated due to incorrect loop bounds, format string attacks, etc. Buffer overflow effects can include overwriting stack memory, heaps, and function pointers.

V. MEASURES TO COPE UP WITH SOFTWARE ATTACKS

The first thing kept in mind while designing countermeasures against software attacks is confidentiality and integrity of data. The most advanced feature of these countermeasures involves governing the accesses of various software components to different portions of the system during different phases of execution, through a combination of hardware and software changes. Since an effective countermeasure must allow the system to provide guarantees about the security of the system starting from the powered-on state, most measures define notions of trust or trust boundaries across the various hardware and software resources. This allows the system to detect violation of trust boundaries and deploys recovery mechanisms. Thus, a trust boundary provides a convenient foundation for the system to make efficient decisions about its security.

Hardware Support- At hardware level security is basically implemented by using secure co-processor module. This processor processes confidential information. Information that needs to be send out of the co-processor is encrypted. Many embedded system also have secure memory areas. These secure memory areas are accessible to trusted system components only.

Secure Bootstrapping- Integrity check could be implemented at boot process level. After power is switched on, system should only be able to access the next layer if all integrity check found
intact. This can be done by comparing the securely saved value
with hashed value of boot process component.

Operating System (OS) Enhancements- Secure Operating Systems provides various features like process isolation, process attestation and secure storage. Secure storage is ensured by using of cryptographic file systems. . To ensure that the software for embedded system meets security requirements, it is important that security should be implemented on various levels.

- **Code and Algorithm Level**-Static analysis tool could be used to scan code to uncover common vulnerabilities.
- **Design and Architecture Level-** System must be coherent and present a unified security architecture that takes into account security principles.
- **Requirements Level-** Security requirement should cover functional security.
- **Hardware Level Security**-At hardware level, security should be implemented on Micro-Architecture Level and at Circuit Level.
- **Micro Architecture Level-** Incorporating security at hardware design of the modules which is specified at the architecture level
- **Circuit Level-** Implementing security at this level means use of techniques at transistor level and package-level to prevent various physical-layer attacks.

VI. CONCLUSION

The evolution of industrial systems is basically based on embedded systems. With increasing interconnection with other industries networks or even the Internet, together with a continuously stronger reliance on open standards such as the TCP/IP protocol suite, creates an increased exposure of automation systems to network-based attacks. consequently, information system security for industrial communication systems is growing in importance. In this paper, we have briefly examined security requirements and challenges in development of embedded systems. We have also examined how an embedded system can be attacked and their counter measures are also examined.

REFERENCES

[1] LAPRIE, J.C. Dependable computing and fault tolerance: concepts and terminology. In Proceedings of the 15th IEEE Symposium on Fault Tolerant Computing Systems (FTCS-15, June 1985). IEEE Computer Society Press, Los Alamitos, CA, 2-11.

[2] Srivaths Ravi , Anand Raghunathan , Srimat Chakradhar, Tamper Resistance Mechanisms for Secure, Embedded Systems, Proceedings of the 17th International Conference on VLSI Design, p.605, January 05-09, 2004

[3] Common Vulnerabilities and Exposures. Available at cve.mitre.org

[4] Latest Virus Threats. Symantec Corporation. Available at http://www.symantec.com/avcenter/vinfodb.html

[5] Virus Information. Computer Security Resource Center, National Institute of Standards and Technology. Available at http://csrc.nist.gov/virus/

Vulnerability notes database. CERT coordination center Available at http://www.kb.cert.org/vuls/

[6] M. Howard and D. LeBlanc. Writing Secure Code. Microsoft Press, 2001.