A UNIFIED VIEW OF EMBEDDED DESIGN SECURITY USING CRYPTOGRAPHIC MECHANISMS

BANAVATHU RAMYASREE[1], VARIKUTI TRIVENI[2]
[1,2]*Assistant Professor, Department of Electronics and Communication Engineering,*
[1,2]*Gurunanak Institute of Technology, Ibrahimpatnam, R.R Dist [India].*

ABSTRACT: Over the course of the past few years, there have been an increasing number of breaches of information security, which has provided a compelling justification for efforts to create secure electronic systems. Embedded systems, which will be used everywhere to capture, store, manipulate, and access sensitive data, present a number of interesting and unique security issues. Security has been the subject of concentrated research in the space of cryptography, registering, and organizing. However, embedded system designers frequently misinterpret security as the addition of features to the system, such as specific cryptographic algorithms and security protocols. In all actuality, it is a completely new metric that creators ought to consider all through the plan cycle, alongside other measurements like expense, execution, and power. The purpose of this paper is to educate embedded system designers and tool developers about the difficulties of creating secure embedded systems. By first examining the typical functional security requirements for embedded systems from the point of view of the end user, we attempt to provide a unified view of embedded system security. After that, we identify the problems that embedded system architects, as well as designers of hardware and software, face, such as creating tamper-resistant embedded systems, meeting security processing requirements, and the effect of security on battery life for battery-powered systems, among other things. In addition, we identify unsolved research issues that will necessitate advancements in embedded system architecture and design methodologies and conduct a survey of potential solutions to these difficulties, drawing on both established practice and emerging research.

KEYWORDS: Embedded Systems, Security, Cryptography, Security Protocols, Security Processing, Design,

I INTRODUCTION

Today, security in some structure is a prerequisite for an expanding number of implanted frameworks, going from low-end frameworks like PDAs, remote handsets, arranged sensors, and savvy cards, to very good quality frameworks like switches, entryways, firewalls, capacity servers, and web servers. The technological advancements that sparked the creation of these electronic systems have also brought about trends in the apparent parallel development of security attacks' sophistication. It has been seen that the expense of weakness in electronic frameworks can be extremely high. For instance, it was assessed that the "I Love You" infection caused almost one billion bucks in lost incomes world wide [1]. According to a survey conducted by Forrester Research [2], a large number of users in the world of mobile commerce (nearly 52% of cell phone users and 47% of PDA users) believe that security is the single largest concern preventing the successful deployment of next-generation mobile services.

Information and communications security has received a lot of attention as the Internet has developed. Secure communications, for instance, make use of a variety of security protocols and standards like IPSec, SSL, WEP, and WTLS [13,23]. While cryptographic algorithms and security protocols address security issues from a functional perspective, many embedded systems are constrained by their operating environments and resources. Security concerns for such systems are shifting from a function-centric to a system architecture (hardware/software) design issue as a result of a number of factors.

The purpose of this paper is to educate embedded[1][2][3] system designers on the significance of embedded system security, examine advancing trends and standards, and demonstrate how the challenges posed by security requirements affect system design. Through the utilization of cutting-edge embedded system architectures and design techniques, novel approaches to resolving these issues will be demonstrated.

II EMBEDDED SYSTEMS SECURITY REQUIREMENTS

Malicious parties may be able to sabotage critical functions provided by embedded systems. It is essential to keep in mind that numerous parties are involved in the typical manufacturing, supply, and usage chain of embedded systems prior to discussing the typical security requirements. Depending on who we take into consideration, the requirements for security can vary.

Take, for instance, a cutting-edge cell phone that enables wireless data, multimedia, and voice communications. When viewed from the perspective of the cell phone manufacturer, the [6][8][17]cellular service provider, the content provider, and the phone's end user, security requirements may differ. An example of this would be the baseband processor. The security of personal data that is stored and communicated by the phone is of the utmost importance to the end user; the copy protection of the multimedia content that is delivered to the phone is of the utmost importance to the service provider; and the secretive nature of the proprietary firmware that is housed within the phone is of the utmost importance to the manufacturer. The set of potentially malicious entities in each of these situations may also differ. For instance, the end user of the mobile device might appear to the content provider to be an unreliable party.

Although this section outlines the broad security requirements that are typical of embedded systems, the specific combination of requirements that apply to each embedded system will be determined by the security model.In the past, communication systems were the first place information security was looked at. It is ideal for two parties to provide security features like data confidentiality, data integrity, and peer authentication when they send or receive sensitive information via public networks or communication channels that are accessible to potential attackers. Confidentiality safeguards sensitive data from unwanted eavesdroppers. Information uprightness guarantees that the data has not been changed misguidedly. Peer authentication makes sure that the information is sent and received by the right people, not by people pretending to be other people. Nowadays, embedded systems used in a wide range of applications must have these security features.

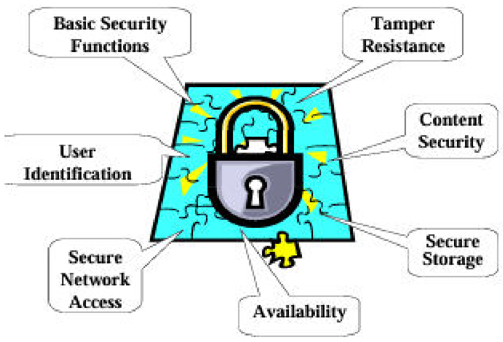

Fig 1 : Security requirements

Fig 1 shows a portion of these security prerequisites according to the viewpoint of an [9][18]end-client, where we utilize the term fundamental security capabilities to signify the arrangement of privacy, trustworthiness and confirmation prerequisites. Very frequently, admittance to the implanted framework ought to be confined to a chose set of approved clients (client distinguishing proof), while admittance to an organization or a help must be given provided that the gadget is approved (secure organization access).

The basic security functions' user or host authentication mechanisms, in addition to biometrics and access control, may be utilized in these.One more fundamental security capability is the accessibility of the embedded system. An embedded system may experience performance degradation or a complete denial of service (DoS) to legitimate users in a number of scenarios when malicious entities prevent it from carrying out its intended functions.When it comes to embedded system security, it is often necessary to safeguard sensitive or critical data or code throughout its lifetime, as well as to ensure that it is properly deleted at the end.

III MECHANISMS TO PROVIDE SECURITY

There are three mechanisms to provide security in embedded systems

a)Symmetric ciphers require the sender to use a secret key to encrypt data (the data being encrypted is often referred to as plaintext) and transmit the encrypted data (usually called the ciphertext) to the receiver. On receiving the cipher text, the receiver then uses the same secret key to decrypt it and regenerate the plaintext. The ciphertext should have the property that it is very hard for a third party to deduce the plaintext, without having access to the secret key. Thus, confidentiality of data is ensured during transmission. Examples of symmetric [13]ciphers include DES, tripleDES, AES, and RC4.

b)Secure Hash algorithms such as MD5 and SHA convert arbitrary messages into unique fixed-length values, thereby providing unique "fingerprints" for messages. Hash functions are often used to construct Message Authentication Codes (MACs), such as HMAC-SHA, which additionally incorporate a key to prevent adversaries who tamper with data from avoiding detection by recomputing hashes.

c)Asymmetric algorithms (also called public-key algorithms), use a pair of keys: one of the keys locks the data while the other unlocks it. Encryption of a message intended for a given recipient requires only the public key known to the world, but decryption is only possible with the recipient's private key, which the recipient should keep secret. [1][3][4]Thus, use of the private key (assuming it is kept secret) provides user or host authentication. Hence, digital signatures are often constructed using public key cryptography and secure hashes. The user can "digitally sign" a message by encrypting a hash of it with his private key, anyone can verify this signature by decrypting with the public key.

Asymmetric ciphers(e.g., RSA, Diffie-Hellman, and so on.) symmetric ciphers, on the other hand, are much faster because they rely on more computationally intensive mathematical functions like modular exponentiation of large integers. Asymmetric ciphers are used to establish (transmit) the secret key for symmetric ciphers across a public network, whereas[20] symmetric ciphers are typically used to encrypt large amounts of data.

Security mechanisms that employ a combination of these cryptographic algorithms within the context of a security protocol are typically used by security solutions to meet the various requirements outlined in the preceding section. In order to provide specific security services, numerous security technologies and mechanisms based on these cryptographic algorithms have been developed.

IV EMBEDDED SYSTEM ARCHITECTURE FOR SECURITY

Traditionally, embedded systems tended to perform a limited number of fixed tasks. The trend is for embedded systems to be able to download new software in order to implement new or updated applications in the field rather than just in the factory, which has a more controlled environment. While this undoubtedly improves an embedded system's flexibility and useful lifetime, it also raises new issues related to the increased likelihood of malicious attacks. An ideal embedded system would provide the necessary security features, efficiently implement them, and defend against malicious attacks. In light of the additional difficulties faced by resource-constrained embedded systems in an environment of ubiquitous networking and pervasive computing, we discuss these below.

Figure 4 delineates the compositional plan space for secure installed handling frameworks. Different full scale engineering models are recorded[11][12] in the first column, and depicted further beneath. Examples of these are the embedded general purpose processor (EP) versus the application-specific instruction set processor (ASIP) versus the EP with specialized hardware accelerators connected to the processor bus, among others. Instruction-set architecture and micro-architecture options for tuning the base processor are detailed in the second row. The security processing features that must be chosen or designed are outlined in the third row. For instance, selecting the functionality that will be implemented using general-purpose instruction primitives, hardware accelerators, or custom instructions. The selection of attack-resistant features for the embedded processor and embedded system design is in the fourth row. As shown in section 4, these safeguard against both physical and software attacks. Process isolation architecture, additional redundant circuitry to thwart power analysis attacks, fault detection circuitry, and an improved memory management unit to manage a secure memory space are examples of this.

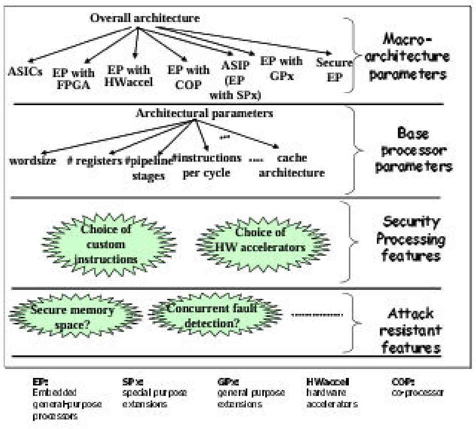
Overall architecture
ASICs
EP with FPGA
EP with HWaccel
EP with COP
ASIP (EP with SPx)
EP with GPx
Secure EP
Macro-architecture parameters
Architectural parameters
wordsize
# registers
#pipeline stages
#instructions per cycle
....
cache architecture
Base processor parameters
Choice of custom instructions
Choice of HW accelerators
Security Processing features
Secure memory space?
Concurrent fault detection?
Attack resistant features
EP: Embedded general-purpose processors
SPx: special purpose extensions
GPx: general purpose extensions
HWaccel hardware accelerators
COP: co-processor

Fig 2 : Security Architecture

V CONCLUSION

In terms of research and widespread use, secure embedded system design is still in its infancy today. The difficulties presented by the process of securing emerging environments or networks of embedded systems necessitate a new approach to the issue, despite the fact that previous research has focused on cryptography, network security, and computer security. The good news is that securing the application-limited world of embedded systems is more likely to succeed in the near future, in contrast to the problem of providing security in cyberspace, where the scope is very large. However, achieving the desired levels of security is significantly hampered by the limited resources of embedded devices.

We believe that we could scale the next frontier of embedded system design, in which embedded systems will be "secure" to the extent required by the application and the environment, with a combination of advancements in architectures and design methodologies. We need to go beyond the basic security features of an embedded system and provide defenses against a wide range of attacks without sacrificing performance, area, energy use, cost, or usability in order to achieve this objective.

REFERENCES

[1] Counterpane Internet Security, Inc. http://www.counterpane.com.

[2] ePaynews- Mobile Commerce Statistics.

http://www.epaynews.com/statistics/mcommstats.html.

[3] W. Stallings, Cryptography and Network Security: Principles and Practice.Prentice Hall, 1998.

[4]B. Schneier, Applied Cryptography: Protocols, Algorithms and Source Code in C. John Wiley and Sons, 1996.

[5]IPSec Working Group. http://www.ietf.org/html.charters/ipsec-charter.html.

[6]SSL 3.0 Specification. http://wp.netscape.com/eng/ssl3/.

[7] Biometrics and Network Security. Prentice Hall PTR, 2003.

[8]OpenIPMP.http://www.openipmp.org.

[9] Internet Streaming Media Alliance. http:/www.isma.tv/home.

[10] MPEGOpen Security for Embedded Systems (MOSES). http://www.crl.co.uk/projects/moses/

[11] Discretix Technologies Ltd. (http://www.discretix.com).

[12] D. Lie, C. A. Thekkath, M. Mitchell, P. Lincoln, D. Boneh, J. C. Mitchell, and M. Horowitz, “Architectural support for copy and tamper resistant software,” in Proc.

ACM Architectural Support for Programming Languages and Operating Systems (ASPLOS), pp. 168–177, 2000.

[13] G. E. Suh, D. Clarke, B. Gassend, M. van Dijk, and S. Devadas, "AEGIS: Architecture for Tamper-Evident and Tamper-Resistant Processing," in Proc. IntlConf. Supercomputing (ICS '03), pp. 160–171, June 2003.

[14] R. M. Best, Crypto Microprocessor for Executing Enciphered Programs. U.S. patent 4,278,837, July 1981.

[15] M. Kuhn, The TrustNo 1 Cryptoprocessor Concept. CS555 Report, Purdue University (http://www.cl.cam.ac.uk/˜mgk25/),Apr. 1997.

[16] G. Hoglund and G. McGraw, Exploiting Software: How to Break Code(http://www.exploitingsoftware.com).Addison-Wesley, 2004.

[17] J. Viega and G. McGraw, Building Secure Software (http://www.buildingsecuresoftware.com).Addison-Wesley,2001.

[18] G. McGraw, "Software Security," IEEE Security & Privacy, vol. 2, pp. 80–83,March–April 2004.

[19] R. Anderson and M. Kuhn, "Tamper resistance- a cautionary note," 1996.

[20] R. Anderson and M. Kuhn, "Low cost attacks on tamper resistant devices," in IWSP: Intl. Wkshp. on Security Protocols, Lecture Notes on Computer Science, pp. 125–136, 1997.

[21] O. Kommerling and M. G. Kuhn, "Design principles for tamper-resistant smartcard processors," in Proc. USENIX Wkshp. on Smartcard Technology (Smartcard '99), pp. 9–20, May 1999.

[22] Smart Card Handbook. John Wiley and Sons.

[23] E. Hess, N. Janssen, B. Meyer, and T. Schutze, "Information Leakage Attacks Against Smart Card Implementations of Cryptographic Algorithms and Countermeasures," in Proc. EUROSMART Security Conference, pp. 55–64, June 2000.

[24] J. J. Quisquater and D. Samyde, "Side channel cryptanalysis," in Proc. of the SECI, pp. 179–184, 2002.

[25] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, "Side Channel Cryptanalysis of Product Ciphers," in Proc. ESORICS'98, pp. 97–110, Sept. 1998.

[26] S. Ravi, A. Raghunathan, and S. Chakradhar, "Tamper Resistance Mechanisms for Secure Embedded Systems," in Proc. Int. Conf. VLSI Design, Jan. 2004.