ISSN: 0975-833X

Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 4, Issue, 04, pp.169-172, April, 2012

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

RESEARCH ARTICLE

HASH BASED HARDWARE MONITORING FOR SECURED EMBEDDED SYSTEM

Sridhar, K. and *Sukandi, A.

Department of E.E.E., S.A. Engg. College, Chennai - 77, INDIA

ARTICLE INFO

Article History:
Received 19th January, 2012
Received in revised form
14th February, 2012
Accepted 07th March, 2012
Published online 30th April, 2012

Key words:

Embedded system security,
Processing monitor,
Security enforcement.

ABSTRACT

The inherent limitations of embedded systems make them particularly vulnerable to attacks. In order to develop a hardware monitor that operates in parallel to an embedded processor and detects any attack that causes the embedded processor to deviate from its originally programmed behavior. To explore several different characteristics that can be used for monitoring and quantify trade-offs between these approaches. This embedded system uses two processor. The failure of one processor activates the other which leads to the continuous function of the system. This project is designed to cater the need of thermal and voltage abnormalities. The abnormalities mentioned above will have lucid way of communication with each other. Both processors are involved in adaptive sharing, the purpose of the proposed method is used in times of only failure of the processor that is processing when the processor fails the particular purpose for only clock cycle that is generated and this flaw is corrected by the second processor. Our results show that our proposed hash-based monitoring pattern can detect attacks within one instruction cycle at lower memory requirements than traditional approaches that use control flow information.

INTRODUCTION

Embedded system finds varieties of application in almost all the sphere human life in many application areas starting from cooking utilities to high end technology like space graft. Generally embedded system are very riskless devices because of its own unique future but often beyond the future some unknown attacks may happened due to various reason like EMI, RFI, harmonics, sag, end of course thermal problems. These problems are not quick common but expected always due to the fast growing multi utilize service like induction machine radio frequency devices rotating machines and much more. When embedded system is utilized for simple work can be rested during the attack but this is not possible in all the cases where like a place can human cannot go and rectifier. Attacks are various kind and doesn't destroy the whole CPU, but disturb flow of activities which in turn comities a malfunction. A malfunction cans a subroutine nearby abnormal proceeding with happened in execution in sequential lines. In this situation for a seamless electronic device which is equal kind of embedded and can perform much more original device is required to overcome the above said. The device utilize a seamless operation will have execution program of its own as well has a program for redundancy operation. Similarly the main device utilize can also have a program of second. In this case both are in networking and can perform indually as well as mutually on requirement basis. Digital computing and communications increasingly pervade our lives, our economy, and our nations' critical infrastructure. Almost everything today is electronic, digital and on-line.

***Corresponding author:** balajisukandi@yahoo.co.in

Security and protection of digital assets is emerging as a discipline of utmost importance. Attacks of the embedded system are classified in many ways as per the nature of attacks in primary nature tamper proof designing will not be so easy due to prohibitive costs incurred in putting together a device that can withstand innumerable, often unknown, attacks, and relentless and rapid improvements in technology constantly, which increase the reach and capability of attackers. In response to this reality, the practical approach is to implement tamper-resistant embedded systems, which translates to tamper-proof for almost all practical purposes.

Attacks are various kinds are discussed briefly

PRIVACY ATTACKS: The objective of these attacks is to gain knowledge of sensitive information stored, communicated, or manipulated within an embedded system.

INTEGRITY ATTACKS: These attacks attempt to change data or code associated with an embedded system.

AVAILABILITY ATTACKS: These attacks disrupt the normal functioning of the system by mis-appropriating system resources so that they are unavailable for normal operation. Software attacks, which refer to attacks launched through software agents such as viruses, trojan horses, worms, etc.

PHYSICAL OR INVASIVE ATTACKS, which refer to attacks that require physical intrusion into the system at some level (chip, board, or system level).

Side-channel attacks, which refer to attacks that are based on observing properties of the system while it performs cryptographic operations, e.g., execution time, power consumption.

PHYSICAL ATTACKS: Physical attacks at the chip level are relatively har to use because of their expensive infrastructure requirements. They can be performed once and then used as precursors to the design of successful non-invasive attacks.

POWER ANALYSIS ATTACKS: The power consumption of any hardware circuit is a function of the switching activity at the wires inside it. Since the switching activity is data dependent, it is not surprising that the key used in a cryptographic algorithm can be inferred from the power consumption statistics gathered over a wide range of input data. These attacks are called power analysis attacks and have been shown to be very effective in breaking embedded systems such as smartcards. Power analysis attacks are categorized into two main classes: Simple Power Analysis (SPA) attacks and Differential Power Analysis (DPA) attacks.

ELECTROMAGNETIC ANALYSIS ATTACKS

The basic premise of many of these attacks is that they attempt to measure the electromagnetic radiation emitted by a device to reveal sensitive information. These attacks against a single chip would require intimate knowledge of its layout, so as to isolate the region around which electromagnetic radiation measurements must be performed. Like power analysis attacks,two classes of EMA attacks, namely, simple EMA (SEMA) and differential EMA (DEMA) attacks.

II.RELATED WORK

Counter measure to take care various attacks:

Attack prevention Attack recovery attack detection ,Tamper evident

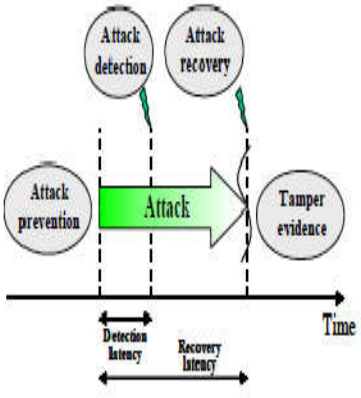

Fig.1.Block Diagram

The term "embedded system" covers a broad range of possible system designs, processor architectures, and performance and functionality characteristics. In our work, we focus on Embedded systems that can be broadly characterized as middle to lower end in the performance spectrum. Their main characteristics are:

1) Medium to low-performance embedded processor core (e.g., single RISC processor);
2) Targeted use for one or only a handful of applications;
3) Typically used in a networked setting.

Examples for practical embedded systems that fit these characteristics are: cellphones, networked sensors, smart cards (typically not networked though), low-end network routers (e.g., home/small office gateway), networked printers, etc.Attacks on embedded system can have a wide range of approaches. Ravi et al. describe mechanisms to achieve physical security by employing tamper-resistant designs [1]. Wood and Stankovic consider a networked scenario where systems are exposed to additional remote attacks [2]. Embedded systems are also susceptible to side-channel attacks (e.g., differential power analysis [3]). Solutions to this problem have been proposed [4], and we do not consider this aspect in our work

III. PROCESSING MONITOR

To achieve secure processing on an embedded system, we use a monitoring system that verifies that the processor indeed performs the operations that it was intended to. For an attacker to abuse an embedded system, it is necessary to modify its operation in some way: either by adding a new piece of instruction code that performs malicious operations or by modifying the existing application to execute malicious code. In this work, we assume that the embedded system workload is "secure" when the applications are executed correctly without any deviation from their binary code. Execution of any instruction that is not part of that binary or that is executed not in the correct order is considered an attack.

3.1. System Architecture

To detect an attack, we employ the system architecture shown in Fig. 1. The system consists of two major components operating in parallel, the conventional embedded processing subsystem and the security monitoring subsystem. The conventional embedded processing subsystem consists of a general purpose processor, memory, I/O, and any other components that are necessary to execute the embedded system application. The only addition is an extension to the processor core that continuously sends a stream of information to the monitoring

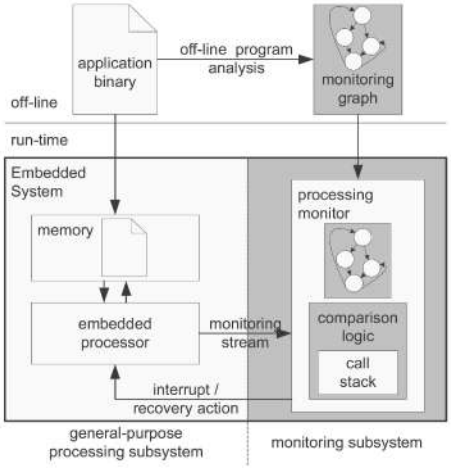

Fig 2.System Architechture

There is also a feedback component from the monitoring system to the processor. In the case an attack is detected, the

monitor can halt the processor and initiate a recovery attempt. The security monitoring subsystem implements the monitoring capability that compares the stream of information sent from the processor with the expected behavior derived from the offline analysis of the binary. A "monitoring graph" represents the sequence of possible control flows between basic blocks. More detailed information about the processing steps within each basic block is also maintained. To be able to keep track of all permissible control flows, a call stack is necessary. If the comparison logic determines that there is a discrepancy between the stream of information from the processor and the monitoring graph, it determines that an attack occurred and initiates an interrupt to the processor. As indicated in the figure, the monitoring graph is generated in an offline process, where the binary of the application is simulated and analyzed. The simulation is necessary to resolve those branch targets that cannot be determined statically. We assume the simplicity of applications used in embedded systems permits a resolution of all branch targets through simulation. This process only requires the binary and not the source code of the application.

IV. RESULTS

We present results on the performance and resource requirements of the proposed monitoring system for the five different information stream patterns. The setup to obtain results is as follows: We simulate the behavior of the monitoring system using an embedded system application workload on an ARM instruction set architecture. We use the MiBench benchmark suite [8] to generate realistic workloads. This suite encompasses over two dozen applications from six different application domains (automotive/industrial, consumer, office, network, security, and telecom). These application domains match very well with the embedded system's complexity that our research targets, and thus, can be considered a representative workload. We employ the SimpleScalar simulator [16] to extract relevant monitoring information and the objdump utility for binary analysis to generate monitoring graphs. A 4-bit hash function is used as a representative of hashed pattern. We pay particular attention to comparing our hash4 monitor to the control flow monitor. The latter is practically identical to the monitoring approach proposed by Arora et al. [6], which is the current state of the art in monitoring in embedded systems. The first important result is that our implementation of the monitoring system performs application monitoring correctly for all applications. That is, no false positives are reported by the monitor when executing the applications on the given benchmark inputs. To quantify the trade-offs between different monitoring patterns, we consider two performance metrics: 1) memory requirement for the monitoring graph and 2) duration of monitoring ambiguity.

4.1 Monitoring Graph Size

The size of the monitoring graph that was generated from the binary of each application is shown in Fig. 3. Each pattern is represented in different shading. We assume a 32-bit address space and an efficient coding of the monitoring graph (e.g., run length coding of sequences of wildcards, efficient coding of consecutive addresses). Each monitoring graph stores the patterns for each basic block as well as the branch pointer at the end of each basic block.

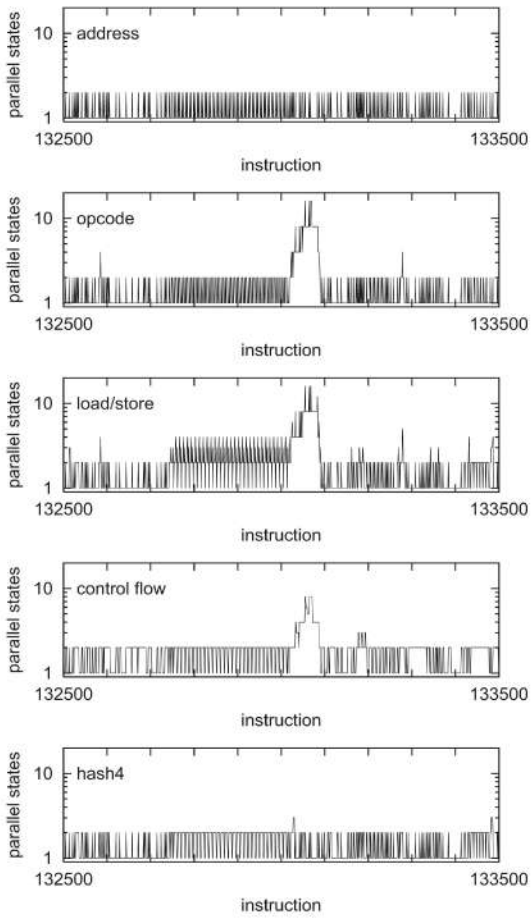

Fig 3. Snapshot of monitoring of 1,000 instructions

V.SUMMARY

In this paper, we have presented an architecture for secure processing in embedded systems. The key idea is to use amonitoring subsystem that operates in parallel with the embedded processor. The monitor verifies that only processing steps are performed that match up with the originally installed application. Any attack would disturb the pattern of execution steps, and thus, alert the monitor.

ACKNOWLEDGEMENT

I extend of gratitude to our beloved chairman Shri.D.Duraiswamy & the Director Shri.D.Venkatesh raja B.E.M.S for keen interest affection shown towards the course. I sincerely thank the Principle Prof.Dr.S.Suyambazalagan M.E.Ph.d(IITM),& grateful thank Mrs.G.Umarani srikanth head of dept. P.G.studies and also thank my internal guide Mrs.S.Sukandi M.E. and my deepfull thank to all non teaching staff members , lovable parents and my sweet hearted friends.

REFERENCES

[1] S. Ravi, A. Raghunathan, and S. Chakradhar, "Tamper ResistanceMechanisms for Secure, Embedded Systems," Proc. 17th Int'l Conf. VeryLarge Scale Integration Design (VLSI Design '04), pp. 605-611, Jan. 2004.

[2] A. Wood and J.A. Stankovic, "Denial of Service in Sensor Networks,"Computer, vol. 35, no. 10, pp. 54-62, Oct. 2002.

[3] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," Proc. 19th Ann.Int'l Cryptology Conf. Advances in Cryptology (CRYPTO '99),

[4] G. Gogniat, T. Wolf, W. Burleson, J.-P. Diguet, L. Bossuet, and R. Vaslin, "Reconfigurable Hardware for High-Security/High-Performance Embedded Systems: The SAFES Perspective," IEEE Trans. Very Large Scale Integration (VLSI) Systems, vol. 16, no.pp. 144-155, Feb. 2008.

[5] D. Arora, S. Ravi, A. Raghunathan, and N.K. Jha, "Secure Embedded Processing through Hardware-Assisted Run-Time Monitoring," Proc. Design, Automation, and Test in Europe Conference and Exhibition (DATE '05),

[6] R.G. Ragel and S. Parameswaran, "IMPRES: Integrated Monitoring for Processor Reliability and Security," Proc. 43rd Ann. Conf. Design Automation (DAC)

[7] M. Abadi, M. Budiu, U. Erlingsson, and J. Ligatti, "Control-Flow Integrity Principles, Implementations, and Applications," Proc. ACM Conf. Computerand Comm. Security (CCS), pp. 340-353, Nov. 2005.

[8] R.G. Ragel, S. Parameswaran, and S.M. Kia, "Micro Embedded Monitoring for Security in Application Specific Instruction-Set Processors," Proc. 2005Int'l Conf. Compilers, Architectures, and Synthesis for Embedded Systems(CASES), pp. 304-314, Sept. 2005.

[9] G.F. Cretu, J.J. Parekh, K. Wang, and S.J. Stolfo, "Intrusion and AnomalyDetection Model Exchange for Mobile Ad-Hoc Networks," Proc. Third IEEE Conf. Consumer Comm. and Networking (CCNC '06), pp. 635-639, Jan. 2006.

****