Design and Implementation of Secure Embedded Systems Based on Trustzone

XU Yan-ling[1] ,PAN Wei[2],ZHANG Xin-guo[1]
1 School of Automation , Northwestern Polytechnical University, Xi'an 710072, China;
2 School of Computer Science , Northwestern Polytechnical University, Xi'an 710072, China
flightxu@163.com

Abstract

Embedded system serves as one of crucial components needed for various applications and services in pervasive computing environment. Security problems related to embedded systems directly influence credibility of these applications and services. In order to effectively eliminate weaknesses in current embedded systems and strongly enhance safety practices of these systems, this paper proposes a Trustzone-based secure enhancement framework for embedded system. This framework consists of a multi-policy access control mechanism and a secure reinforcement method. The multi-policy access control mechanism establishes multiple secure policies by utilizing the Domain and Type Enforcement (DTE) model and an improved Bell-La Padula (BLP) model, and the secure reinforcement method provides powerful safeguards through the employment of Linux Security Module (LSM) framework. We construct a secure embedded system environment based on TrustZone technique and secure Linux system. A prototype system founded on ARM Linux achieves rational combination of secure operating system and trustworthy hardware techniques and thus ensures diversified applications and services safety.

1 Introduction

Nowadays diversified network computing resources and communication systems that apply in such aspects as quotidian living, economic practices, as well as governmental infrastructures, and the like, necessarily rely on the development and application of embedded technology. Serving as supporting components in pervasive environment, embedded systems are playing an important role on various applications and services. However, security problems related to embedded systems make them counterproductive, especially when underlying safety precautions are of absence in open networking surroundings.

These problems concerning security of embedded systems involve in three aspects: confidentiality, integrity and availability[1]. Confidentiality of system is to prevent sensitive system information from unlawful access and intentional abuse; integrity of system is to guarantee critical files and data against deletion and modification in unauthorized ways, and provide tamper-proofing protection for applications and services on embedded operating system from malicious code and virus program; availability of system is to defend whole system against hostile attacks and ensures authorized, legitimate access. The reasons that current embedded systems have suffered a myriad of penetration and threat attribute to intrinsic weakness of hardware structure and unassured status of operating system. Though many security solutions claim that such technique as IDS, firewall, etc, can carry out secure reinforcement, internal vulnerabilities seriously impair external effectiveness these solutions play on operating system and expose whole system to malicious and illegal communities. Therefore, embedded system fails to guard applications and services based on it. But with the maturity of trustworthy computing technique, a novel scheme becomes possible in effectively solving deficiencies of computer architecture and strongly enhancing security of embedded systems. Thus depending on trustworthy hardware and secure operating system, a viable and potent security framework is presented in our investigation.

2 Related technical analysis

2.1 Trusted hardware technology

The security solutions based on trustworthy hardware technique mainly include AEGIS secure processor presented by research group in Massachusetts Institute of Technology University, eXecute Only Memory technique proposed by research team in Stanford University, and TrustZone secure

978-0-7695-3287-5/08 $25.00 © 2008 IEEE
DOI 10.1109/ICESS.2008.59

processor architecture designed by ARM corporation [2].

AEGIS is a secure startup architecture that enables multilevel verification from system startup to application[3]. The AEGIS architecture provides many applications like commercial grid computing, software authorization and digital copyright management, and so on. AEGIS secure processor technique is successful to enhance creditability of central processor units.

XOM, the abbreviation of eXecute Only Memory, is another secure architecture[4]. In this architecture, XOM supposes that internal units at processors defend against any attack and applications place trusts on CPU rather than whole operating system., XOM architecture provides powerful solutions to prevent from software piracy and other attacks.

TrustZone technique implements zone isolation by a unique secure zone, i.e., trusted zone[5]. Trusted zone serves as an essential component to establish the connection between user mode and kernel mode and has a higher privilege level than kernel mode. Trusted zone is not an exclusive zone, but a zone where kernel mode or user mode applications run. Monitor module is an encapsulated, unrepeatable utilizing program, which is independent to operating system. This module controls switching between normal zone and trusted zone, protects the switching context, and supervises all tasks in processors in real time. The core of trusted zone ensures data access safety and integrity. If a secure request is captured, the encryption of this request is completed in normal zone and then stored in sharing part assigned by secure kernel. Once this request is verified, monitor module records non-secure states and then switches monitoring sessions to trusted zone. Monitor module can protect data in trusted zone from infiltrating into normal zone. All these practices are achieved by hardware technique not by operating system. The TrustZone architecture is shown as Figure1.

TrustZone technique develops a high level software architecture supported by hardware protection. This architecture provides secure hardware base for many operating system such as Palm OS, embedded Linux, Symbian OS, and Windows CE, and the like. Specially, Compare with AEGIS and XOM, the TrustZone architecture has been widely recognized as open secure architecture and trusted computer base and successfully used in embedded systems needed for high secure level. In this paper, we present a TrustZone-based secure enhancement framework for embedded system.

2.2 Embedded Linux system security

Open source Linux systems have served as the leading platforms for the development of embedded systems. Prevailing embedded Linux systems include: μCLinux that manages microcontrollers without Memory Management Units (MMUs), RTLinux that supports real time applications, and ARMLinux that administers microcontrollers with MMUs, and so on. An embedded Linux kernel is a derivative of Linux kernel and inherits most advantages from Linux operating system. Embedded systems successfully provide many applications such as routers, STB, PDA, intelligent appliances, instruments and facilities for aeronautic and astronautic researches.

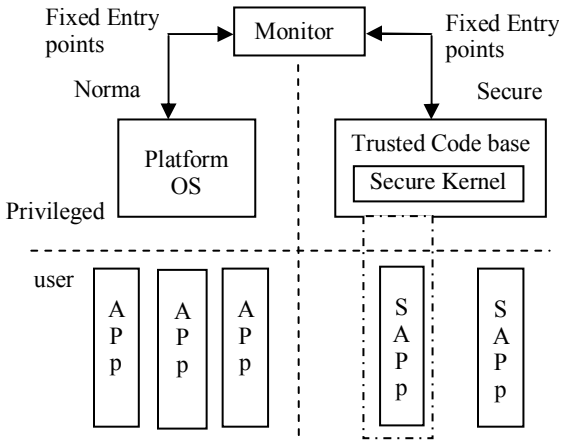

Figure 1: TrustZone architecture

In embedded Linux systems, operating system security determines security of various applications and whole embedded systems. Access control mechanism plays an essential role for operating system security. In an operating system, vulnerability of access control mechanism is the main reason that causes threat of confidentiality and integrity. Therefore, access control is the main content for security mechanism of operating system. Discretionary access control (DAC) adopted in embedded Linux is a simple access control mechanism. Security properties of objects and access control information are assigned by the process they belong to. Advantage of this mechanism is that it's quite flexible. However, the access control is easy to transfer and is prone to attacked by malicious programs such as Trojan horse. Considering such deficiency of access control mechanism in current Linux system, embedded platform security cannot be assured merely through embedded Linux operating system itself and certain security software. Therefore, to improve security of embedded system, we have to study access control technology with security enhancement.

3 Secure enhancement model for embedded Linux system

3.1 Secure embedded system framework based on TrustZone

In a TrustZone-based embedded system, if an application in kernel mode requests to visit trusted zone, operating system need verify the security of this application. If the verification is passed, the application calls Secure Monitor Interrupt (SMI) instructions to visit trusted zone. As for trusted zone, whether the trusted zone is safe or not lies on secure attributes of operating system kernel. When SMI instructions are performed, security tags in specific registers of SMI instructions are marked to identify the application, and then operating system switches operations to trusted zone. Secure kernel in trusted zone has rights to scrutinize the verification of applications running in it. As long as the values of security tags are not changed, monitor module administers all operations in processors, performs secure switching between normal zone and trusted zone. In a system built by TrustZone-based secure embedded system framework, secure switching between normal zone and trusted zone is a foundation of operating system security. Because general applications need not perform in trusted zone, security of general applications cannot be protected by TrustZone technique effectively. Consequently, the enhancement of embedded operating system security is an essential key to guarantee the implementation and application of TrustZone technique.

3.2 Multi-policy mandatory access control mechanism

In most applications, security of embedded system means the basic security requirement of operation system, that is, confidentiality and integrity. Mandatory access control mechanism (MAC) can ensure confidentiality and integrity of system. It's the main security enhancement measure adopted by most operating system. Domain and Type Enforcement (DTE) model and Bell-La Padula (BLP) model are two usual security models to enhance access control. DTE model provides a mandatory access control with the same security level as that of BLP model. Based on DTE model and BLP model, the author puts forth a joint mandatory access control mechanism. Integrity is ensured by using of DTE model and confidentiality is ensured by using of improved BLP model.

DTE is an access control model based on table and it can also implement integrity independent of trusted users[6]. In DTE model, all subjects or processes in the system connect with a domain and all objects or resource connect with a type. DTE firstly establishes a domain definition table, describing operation right of each domain on different types of resource. And a domain interaction table is established to describe the allowed access modes between domains. According to domain and resource type to access, the system searches in domain definition table to decide whether access is allowed or not. If an access request is allowed, then the process can access the needed resource, vice versa. Access control configuration in DTE model is quite flexible and fine-grained adjust of access right of process is allowed to minimize effect of a given process on system and breakage of malicious attack on the system. Buffer overflow attack can be prevented to protect integrity of crucial resource within the system.

BLP model is a state machine model[7]. It simulates security model of computer operating system that meets the military security standard. In the model, system state and transition rule between state are defined and security axiom that control system state transition rule.

State of BLP model is a four-tuples set（B, M, f, H）. Here B denotes the current accessing set, which is composed of a three-tuples set (S, O, A). S denotes the process set, O denotes object set and A denotes access attribute set. A={r, a, w, e}, where r denotes " read only", a denotes "append", w denotes" write" and e denotes "execute". M denotes access control matrix, f is security function used to differentiate security level between subjects and objects. H denotes hierarchy of objects. Security level of subjects includes the maximal level and the current level. Cs denotes security level of subjects and Co denotes security level of objects.

BLP model groups entities within the system into different access levels by classifying subjects and objects into different levels and categories. Thus confidentiality of system is ensured. The following properties and rules are kernel of BLP model.

1. Simple-security property

If the current access B = (S, O, r), then Cs ≥ Co where sign ≥ denotes "more than".

2. *-property

At any situation, Access of subjects to objects must meet the following standard:

If the current access B = (S, O, a), then Cs ≤ Co.

If the current access B = (S, O, r), then Cs ≥ Co.

If the current access B = (S, O, w), then Cs = Co.

BLP policy can effectively prevent Trojan horse from intrusion and further damage.

3. Improvement of integrity policy of BLP model

In multi-security policy model of BLP, we modify integrity of BLP model as following:

(1) if Cs ≥ Co
then B = (S, O, r) or B = (S, O, e);
(2) if Cs = Co
then B = (S, O, w) or B = (S, O, a).

Besides confidentiality of BLP model, the "reading up" of BLP is added with integrity requirement. By restriction of "over writing", only append mode is allowed to implement "reading up" to prohibit covert channel.

3.3 Multi-policy mandatory access control kernel logic based on LSM

Embedded Linux operating system usually runs in single user mode. A user's operation is executed by a process. A process is the only secure subject and objects include all objects within the operating system. With Linux Security Module (LSM) framework's imbedding into Linux2.6 kernel, there exits a uniform measure for implementation of mandatory access mechanism[8]. embedded Linux operating system with security enhancement uses LSM framework, adopts security policy of BLP model and DTE model, utilizes security module stacking technology and assigns security label for process and resource in the system to implements mandatory access control. The security label can be defined as a two-element array (domain/type: security level), where domain denotes sign of users' privilege of process operation, type denotes class sign of resource, security level denotes security level of use process and objects. Domain/type implements security policy of DTE. The system control access resource by domains and types to ensure integrity of the system. Security level implements security policy of BLP to improve confidentiality of system. Therefore, the reinforced operating system ensures secure implementation of TrustZone architecture. Multi-policy mandatory access control based on LSM and its kernel logic are illustrated in figure 2.

3.4 Division of domain and type and division of security level

Division of domain and type is essentially definition of domain, type, transition rule of domain and default domain according to users' requirements. It's the prerequisite to design global setup file of DTE so as to control access precisely. Security level is foundation for implementation of BLP model. Division of domain and type and division of security level can be described as following:

1) Division of domain

According to security requirement and running environment of embedded system, DTE policy can be divided into 6 categories:

daemon_d: a domain for system daemons and its initial daemons.

login_d: a domain for LSM security module-related daemons. Login daemons must follow LSM security policy.

user_d: a domain for general application daemons.

system_d: a domain for system operating user session.

trusted_d: a domain for secure application daemons and trusted daemons.

network_app_d: a domain for network application daemons.

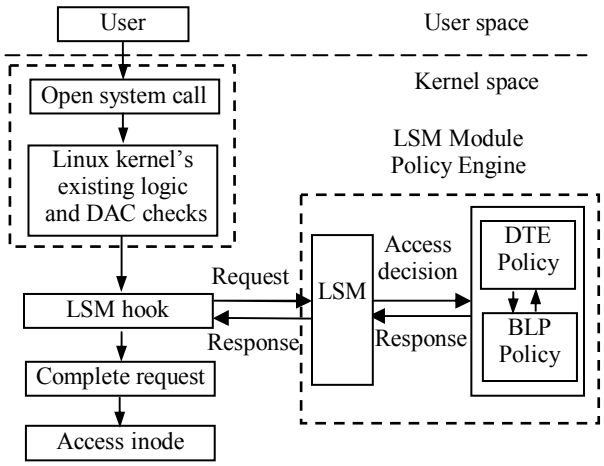

Figure 2: Multi-policy mandatory access control kernel logic based on LSM

2) Division of types

All the objects is divided into 7 types according to DTE policy as following:

base_t, basic type.

bin_t，system binary file type.

conf_t，system configuration file type.

usr_t，users' file type.

trust_t，file type of trusted process and secure application process.

secpolicy_t，security policy file type.

network_t, network application file type.

3) Division of security level

Security level of subjects can be defined as a six-element array Cs= {Top Secret, Secret, confidentiality, Classified, Unclassified, anonymous}, where "anonymous" is reserved for access of shared objects in the domain. Security level of objects can be defined as a six-element array Co= {Top Secret, Secret,

confidentiality, Classified, Unclassified, Shared}, where "shared" is reserved for objects that is accessed by anonymous process.

4 Secure embedded system implementation

4.1 Secure embedded system architecture

The secure embedded system architecture based on TrustZone technique and secure Linux operating system is shown as figure 3. General applications belonging to normal zone run on secure embedded Linux system. Normal zone allocates BLP policies and DTE policies to avoid malicious trespasses, and thus assures confidentiality and integrity of whole system. Secure applications call TrustZone access control driver and SMI by performing trusted processes. Monitor module establishes secure switching between normal zone and trusted zone to scrutinize any access to trusted zone. Trusted applications belonging to trusted zone directly function on secure kernel, which is supported by TrustZone technique to achieve necessary protection and access control for applications. This architecture can provide secure execution environment for open embedded systems and various applications. In our research, we develop a prototype system founded on ARM Linux with Linux kernel 2.6.18 and embedded processors with kernel ARM1176JZF-S.

4.2 Access control policy within multi-policy mandatory security framework

Steps for subjects to access objects within the improved security framework are as following:

(1) Subject S requests access to object O via system call;

(2)The system call requests decision of both DAC model and LSM model to tell if S can access O;

(3)If DAC doesn't permit access of S to O, then an error message is returned and jumps to step (5); If DAC permits access of S to O, then DTE of LSM module continues to make decision. If DTE doesn't permit access of S to O, then an error message is returned and jumps to step (5). If DTE permits access of S to O, then checks Cs and Co based on BLP policy. If BLP doesn't permit access of S to O, then an error message is returned and jumps to step (5); if BLP permits access of S to O, a correct message is returned;

(4) DAC and LSM return the result to the system call;

(5) The system call returns an error message and doesn't permit access of S to O;

(6)S accesses O.

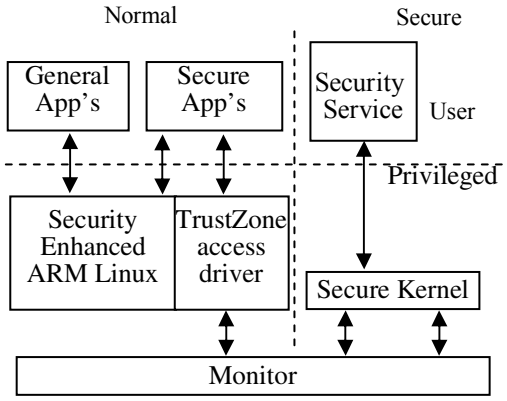

Figure 3: Secure embedded system architecture

5 System security analysis

5.1 Application security in normal zone

The protection of secure applications in normal zone is an essential point for secure embedded system. In order to guarantee security of these secure applications, normal zone employs DTE policies to set value as trust_t for TrustZone access control driver and SMI, and set Trusted_d for invoked processes that include SMI instructions and secure application that request to enter trusted zone. When requesting to enter trusted zone, a secure application in normal zone sends a request, and then operating system call SMI instructions. The validity of applications is verified by operating system. According to DTE policies, none but a secure application that has been valued as Trust_d can call SMI instructions and TrustZone access control driver, and functions in trusted zone. Through allocating DTE policies, secure switching between normal zone and trusted zone can be achieved successfully, and potential risks traditional secure switching mechanisms have suffered can be avoided in an effective way.

5.2 Whole system security and performance impact

Confidentiality and integrity of our prototype system is guaranteed by LSM. LSM protects not only general applications but also secure applications in normal zone. Because LSM provides mandatory access control, general applications just function in normal zone rather than in trusted zone, and system consumption can be efficaciously saved. We compared a standard Linux 2.6.18 kernel against a enhanced

security 2.6.18 kernel with the LSM, we used the LMBench to measure the performance of core kernel system calls and facilities. The worst case overhead was 9.4% for open/close, and 11.8% for file delete. Meanwhile, in order to protect secure applications belonging to normal zone, our prototype system employs SMI instructions to manage secure switching between normal zone and trusted zone. Because of trusted hardware technique, system consumption needed for secure switching can be also efficaciously minimized. In sum, our prototype system successfully achieves rational combination of secure operating system and trustworthy hardware techniques, thereby ensuring diversified applications and services safety

6 Conclusion

With the development of pervasive computing, embedded systems have been widely used in such fields as network communication, e-business, consumption electronics, industry control, etc. In order to provide secure embedded computing environment for various applications and services, the construction and development of secure embedded architecture becomes a crucial focus. This paper employs mandatory access control to operate embedded Linux system on level B1 security standard, and presents an embedded system security solution based on TrustZone technique and secure embedded Linux. The proposed solution can serve as a viable and effective candidate for settling security problems in embedded systems.

Acknowledgements

Our research was funded by the National High Technology Research and Development Program of China (863 Program) under agreement number 2006AA01Z406.

References

[1] S.Ravi, A.Raghunathan ,S.Chakradhar,Tamper Resistance Mechanisms for Secure Embedded Systems, Proceedings of the 17th International Conf on VLSI Design (VLSID'04), Mumbai,India,2004,605-611.

[2] Secure architecture in embedded systems: an overview ReCoSoc'06,Montpellier,France,2006. http://hal.archives-ouvertes.fr/docs/00/12/44/12/PDF/recosoc06.pdf.

[3] G. E. Suh, D. Clarke, B. Gassend, M. van Dijk, and S. Devadas. AEGIS: Architecture for Tamper-Evident and Tamper-Resistant Processing. In Proceedings of the 17th Int'l Conference on Supercomputing (MIT-CSG-Memo-474 is an updated version), June 2003.

[4] D. Lie, C. Thekkath, M. Mitchell, P. Lincoln, D. Boneh, J. Mitchell, and M. Horowitz. Architectural Support for Copy and Tamper Resistant Software. In Proceedings of the 9th Int'l Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX), pages 168–177, November 2000.

[5] T. Alves and D. Felton. TrustZone: Integrated hardware and software security. ARM white paper, July 2004.

[6] S.E. Hallyn., P.Kearns.,"Domain and type enforcement for Linux", Proceedings of the 4th Annual Linux Showcase and Conference, Atlanta,Georgia,USA,2000,247～260.

[7] Bell D E, Lapadula L J.Secure Computer System: Unified Exposition and MULTICS Interpretation. MTR-2997 Rev. 1, The MITRE Corporation, Bedford, MA, USA, Mar.1976.

[8] C.Wright, C.Cowan, J.Morris et al. "Linux Security Module Framework", http://www.kroah.com/linux/talks /ols_2002_lsm_paper/lsm.pdf.