Reliable Computing Under Resources Constraints Policy

[1]S. Anitha Reddy, [2]V. Uma Maheswari
[1]Dept. of CSE, Krishna Murthy Institute of Technology and Engineering, AP, India
[2]Dept. of CSE, Vardhaman College of Engineering, AP, India

Abstract
Hardware-based trusted computing platforms are intended to overcome many of the problems of trust that are prominent in computing systems. In this paper, a result of the Software Engineering Institute's Independent Research and Development Project "Trusted Computing in Extreme Adversarial Environments: Using Trusted Hardware as a Foundation for Cyber Security," we discuss the capabilities and limitations of the Trusted Platform Module (TPM). We describe credential storage, device identity, chains of trust, and other techniques for extending hardware-based trust to higher levels of software-based infrastructure. We then examine the character of trust and identify strategies for increasing trust. We show why acceptance of TPM-based trust has been limited to date and suggest that broader acceptance will require more focus on traditional trust issues and on end-to-end services..

Keywords
Reliable Computing, Symmetric Cryptographic Primitives, Under Policy

I. Introduction
Trustworthiness is a measure of the integrity, ability, competence, and surety of an entity to provide a service. As the number of security risks in automated and networked systems continues to rise, assurance of the trustworthiness of systems becomes increasingly important to safely and cost effectively use services involving automation or networks.
Trust is the degree of confidence (i.e., certainty, reliance, or belief) that one has in the trustworthiness of a person, organization, application, or system to satisfy an individual's expectations in providing a particular service. The degree of confidence depends not only on the trustworthiness of the service and its provider, but also on the user's knowledge of that trustworthiness. One should trust only the trustworthy, but that is impossible without evidence of their trustworthiness.
Issues of trust and of adequate evidence of trustworthiness are complex, difficult, and until recently, seldom addressed by the developers of automated and network systems.
They are, however, a critical aspect of everyday human life. Trust involves many risks that must be managed. Automated systems and networks introduce additional trust issues, primarily in the area of security, that are uncommon in social systems alone. The use of authenticators (passwords, tokens, and biometrics), digital signatures, crypto checksums, reputation systems, and digital identity management methods, for example, are security mechanisms that are sometimes necessary for trust in automated and networked systems, but do not address the trustworthiness of the service providers nor the functionality or quality of the services themselves. Hardware-based trusted computer platforms are intended to overcome many of the problems of trust that are prominent in computing systems [Smith 2005], [England 2004]. The TPM is a simple, passive, integrated circuit chip intended to provide several trustworthy security capabilities, to be extensible to higher levels of computer software and network infrastructure, and to create a foundation for trusted applications. In practice, however, hardware-based trusted computer platforms are rarely used to develop computational or network infrastructure and have had limited market success in end applications to date, even when the TPM chip is present.
Many emerging applications will rely on extensive mutual co-operation among a highly interconnected network of computers . A group of computers working together may decide the setting of a thermostat based on weather forecasts received directly from computers in the local weather station. Computers in a car, interacting with computers in cars nearby may decide the best course of action to avoid an impending collision. Sensors monitoring vital internal organ functions of a person on the road may relay early warning signs over multi hop ad hoc networks to the nearest hospital to facilitate timely responses.
In such applications, each device is expected to perform some tasks for the overall good of the network. An obvious requirement in such scenarios is the ability to trust the devices. It does not take much imagination to see the consequences of an attacker's ability to impersonate a sensor to send a false alarm or a malicious course correction.

II. Literature Review
Research on the use of tamper-resistant hardware has been in progress for nearly 15 years public concerns for issues such as copy protection and secure remote execution and the recent push in commodity secure hardware suggest that the benefits of using secure hardware is now exceeding its overhead in complexity, performance, and cost. This project describes a trusted computing architecture, Cerium, that uses a secure processor to protect a program's execution, so that a user can detect tampering of the program's instructions, data, and control-flow while the program is running. This project considers the following computation model. A user runs a program on a computer outside the user's control. The computer runs the program and presents the user with an output. The user wants to know if the output is in fact produced by an un-tampered execution of the user's program. We call this computation model tamper-evident execution. Tamper-evident execution enables many new useful applications. For example, a project that depends on distributed computation, such as SETI@ home, can use tamper-evident execution to check that results returned by participants are produced by the appropriate SETI home software. The goal of Cerium is to support tamper-evident execution while facing strong adversaries. At the user level, Cerium should expose malicious users forging results of other users' programs without running them. At the system level, Cerium should expose buggy operating systems that allow malicious programs to modify the instructions and data of other programs. At the hardware level, Cerium should detect hardware attacks that tamper with a program's data while they are stored in memory, such as attacks on the DRAM or memory bus. Such strong adversaries prevent us from using software only techniques to implement tamper-evident execution.Smart cards are increasingly prevalent, particularly in Europe, for authentication and payment mechanisms (credit cards, pay-TV access control, public transport payment, medical records, personal identity, mobile phone SIMs, etc.). They present a harder target for the criminal underworld than

their magnetic strip counterparts. None the less, there is suf_cient economic gain in cracking smart cards. Pay-TV is particularly vulnerable since communication with the smart card is typically unidirectional, from the broadcasting source to the set-top box hosting the smart card. Since there is no back channel, it is not possible to identify duplicate smart cards via interactive protocols. Consequently, it is economically attractive to reverse engineer a pay-TV smart card in order to make a large number of duplicates. As smart cards are used in more and more applications, many new opportunities for theft and fraud open up to criminals capable of reverse engineering cards or extracting key material.
The next section introduces attack technologies which determine the environment in which smart cards must survive. We address a number of hardware level security issues and how self-timed circuits can be used to build more robust smart cards.
Technique used or algorithm used :

- UNDER policy
- Encryption
- Decryption

A. Advantages
Mandates sufficiently trustworthy computers that can be realized at low cost . The often heard statement that "complexity is the enemy of security" is far from dogmatic. For one, lower complexity implies better verifiability of compliance. Furthermore, keeping the complexity inside the trust boundary at low levels can obviate the need for proactive measures for heat dissipation. Strategies constrained to simultaneously facilitate shielding and heat dissipation tend to be expensive. On the other hand, unconstrained shielding strategies can be reliable and inexpensive to facilitate.

B. Applications
In emerging application scenarios, calling for very large scale deployments of inexpensive devices bound to low complexity ScPs, the use of asymmetric cryptographic primitives may not be feasible. It was argued that low-complexity ID-based schemes are very much desirable for many emerging applications scenarios.

III. Existing System
In case of the existing system each and every system are considered as a trusted computer. And so the attacker finds it easy to attack the system with fake signals. And also in the emerging network where many are used for some good propos. And in those there a lot of chance for the attacker to send unwanted information. In case of the fire alarm, if all the system are considered as trusted they could send false alarm where it lead to a heavy loss. And so we need a system to protect it. Hence we develop a new system.

IV. Proposed System
The proposed system we introduce a new technology to protect the network. This is achieved by the following way. Realizing widespread adoption of such applications
Mandates sufficiently trustworthy computers that can be realized at low cost. Apart from facilitating deployment of futuristic applications, the ability to realize trustworthy computers at low cost can also addresses many of the security issues that plague our existing network infrastructure. Although, at first sight, "inexpensive" and "trustworthy"
May seem mutually exclusive, a possible strategy is to reduce the complexity of the components inside the trusted boundary. The often heard statement that "complexity is the enemy of security" is far from dogmatic. For one, lower complexity implies better verifiability of compliance. Furthermore, keeping the complexity inside the trust boundary at low levels can obviate the need for proactive measures for heat dissipation. Strategies constrained to simultaneously facilitate shielding and heat dissipation tend to be expensive. On the other hand, unconstrained shielding strategies can be reliable and inexpensive to facilitate.

V. System Design

A. Input/ Output
The input will be choosing trusted system and selecting IP address of both the trusted and non-trusted systems if there is no stored IP then new IP address will be entered and the output will be IP address gets stored in database and direct us to the main form.

1. Modules
- LOGIN MODULE
- TRUSTED SYSTEM MODULE
- CRYPTOGRAPH
- Encryption
- Decryption
- SENDING MODULE
- RECEIVING MODULE

2. Module Description

B. Login Module
User gives the required username and password and then logins. If the login name and password in correct then he goes to the next form else he is asked to give the correct username and password.

VI. Trusted System Module
Any trusted computer defines a clear trust boundary. For example, for a single chip ScP all components inside the chip may fall under such a trust boundary. Enforcing the trust boundary is by proactive measures for protection of components within the boundary. However, the regions inside a trust boundary that are physically protected can change dynamically, depending on the state of the ScP. when the CPU is off, there is no need to extend protection to all regions. However, when the CPU is on, the scope of protection will need to be wider.

VII. Cryptography

A. Encryption
In this module, we investigate the suitability of UNDER for Identity-Based Encryption (IBE) and signature (IBS) schemes. We then motivate the need for low complexity ID-based authentication schemes for ScPs for evolving application scenarios. This includes an overview of some existing low-complexity ID-based KPS

B. Decryption
In this module a private exponent d is used for decryption and signing. More specifically, the private exponent needs to be stored in RAM for performing computations like decryption and signing. Modular exponentiation is often performed using the square-and-multiply algorithm.

VIII. Sending Module
In this module, the encrypted file is sent to the non-trusted system with the key, normal file is sent to the trusted system and also

read only files are sent while sending the files details about the file and the path of the file is stored in data base. Before sending the file to the trusted and non-trusted systems we have to make sure that the server is made to run so that it can receive files from the client.

IX. Receiving Module

In this module the files are received. If it's a trusted system then the files receives without decryption else it receives in encryption mode with a secret key to decrypt the encrypt file and view the file. The file are usually stored in the path "c:\receive". If it's a read only file the user cannot edit or modify the file.

X. System Testing and Maintainance

A. Testing

Software Testing is the process used to help identify the correctness, completeness, security, and quality of developed computer software. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors. Quality is not an absolute; it is value to some person. With that in mind, testing can never completely establish the correctness of arbitrary computer software; testing furnishes a criticism or comparison that compares the state and behavior of the product against a specification. An important point is that software testing should be distinguished from the separate discipline of Software Quality Assurance (SQA), which encompasses all business process areas, not just testing. There are many approaches to software testing, but effective testing of complex products is essentially a process of investigation, not merely a matter of creating and following routine procedure. One definition of testing is "the process of questioning a product in order to evaluate it", where the "questions" are operations the tester attempts to execute with the product, and the product answers with its behavior in reaction to the probing of the tester[citation needed]. Although most of the intellectual processes of testing are nearly identical to that of review or inspection, the word testing is connoted to mean the dynamic analysis of the product—putting the product through its paces. Some of the common quality attributes include capability, reliability, efficiency, portability, maintainability, compatibility and usability. A good test is sometimes described as one which reveals an error; however, more recent thinking suggests that a good test is one which reveals information of interest to someone who matters within the project community.

B. Maintenance

The objectives of this maintenance work are to make sure that the system gets into work all time without any bug. revision must be for environmental changes which may affect the computer or software system. This is called the maintenance of the system. Nowadays there is the rapid change in the software world. Due to this rapid change, the system should be capable of adapting these changes. In our project the process can be added without affecting other parts of the system. Maintenance plays a vital role. The system liable to accept any modification after its implementation. This system has been designed to favor all new changes. Doing this will not affect the system's performance or its accuracy.

XI. System Implementation

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Implementation is the process of converting a new system design into operation. It is the phase that focuses on user training, site preparation and file conversion for installing a candidate system. The important factor that should be considered here is that the conversion should not disrupt the functioning of the organization.

XII. Scope for Future Hancements

The project has covered almost all the requirements. Further requirements and improvements can easily be done since the coding is mainly structured or modular in nature. Improvements can be appended by changing the existing modules or adding new modules. One important development that can be added to the project in future is file level backup, which is presently done for folder level.

XIII. Conclusion

It is concluded that the application works well and satisfy the needs. The application is tested very well and errors are properly debugged. It also acts as the managements Systems to the valuable resources.

In this project student data entry module which allows admin to enter the details of student and their contact details. The report is generated for each student serial no. It also as unique search facility for mobile and land no through this particular student details are viewed immediately.

References

[1] M. Kwiatkowska, V. Sassone,"Science for Global Ubiquitous Computing", Grand Challenges in Computing (Research), T. Hoare and R. Milner, eds., 2004.

[2] S.W. Smith,"Trusted Computing Platforms: Design and Applications", Springer, 2005.

[3] R. Anderson, M. Bond, J. Clulow, S. Skorobogatov, "Cryptographic Processors—A Survey", Computer Laboratory Technical Report UCAM-CL-TR-641, Univ. of Cambridge, Aug. 2005.

[4] R. Anderson, M. Kahn,"Tamper Resistance—A Cautionary Note", Proc. Second Use nix Workshop Electronic Commerce, pp. 1-11, 1996.

[5] S.W. Smith, S. Weingart,"Building a High-Performance Programmable Secure Coprocessor", Computer Networks, Vol. 31, pp. 831-860, 1999.

[6] R. Needham, M. Schroeder,"Using Encryption for Authentication in Large Networks of Computers", Comm. ACM, Vol. 21, No. 12, Dec. 1978.

[7] D. Lie, C.A. Thekkath, M. Horowitz,"Implementing an Untrusted Operating System on Trusted Hardware", Proc. 19th ACM Symp. Operating Systems Principles, pp. 178-192, Oct. 2003.

[8] P.C. van Oorschot, A. Somayaji, G. Wurster,"Hardware-Assisted Circumvention of Self-Hashing Software Tamper Resistance", IEEE Trans. Dependable and Secure Computing, Apr. 2005.

[9] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, T. Rabin,"Tamper Proof Security: Theoretical Foundations for Security Against Hardware Tampering", Proc. Theory of Cryptography Conf.,Feb. 2004.

[10] S. Weingart,"Physical Security for the mABYSS System", IEEE Security and Privacy, pp. 38-51, 1987.

S. Anitha Reddy, Associate Professor, Department of Computer Science and Engineering, Krishna Murthy Institute of Technology and Engineering, Edulabad, Ghakesar.

V. Uma Maheswari, Assistant Professor, Department of Computer Science and Engineering, Vardhaman College of Engineering, Shamshabad, Hyderabad, India.