Global Information Assurance Certification Paper

Trusted Computer Systems: Understanding and Issues

Ahmad Ubaidah Omar
SANS Security Essentials GSEC Practical Assignment Ver 1.3
25 March 2002

Abstract

The ever growing reliance of world communities on computer based systems has pushed their security issues into the primetime. But the very nature of those systems are very complex and sometimes involve too abstract concepts for straightforward comprehension. Therefore it is very naive to say that a system is secure whilst it is still not well understood. Instead a degree of trust must be assigned to the security features and assurance of the system. In other words, security of a system is trusted up to a certain level of measurable assurance. This paper will discuss the concepts of trusted computer systems and highlight some reasons of why we need to use it. Several available implementations will be discussed and continue with products in development or research. Next some issues that hinder the successful adoption of the systems on worldwide scale will be highlighted and finally suggest some ideas on how to overcome the issues.

1. Introduction

Computer based systems or computer systems are the main building blocks of almost all of the information systems in the world nowadays. These systems typically consist of hardware and software. It is fair to say that the software side is the intelligent and most complex element that drives these systems. The underlying nucleus of this 'soft' side is known as operating system. Operating system and all of the components that form the computer systems are normally very complex and subjected to various scrutiny and study in terms of security implementations. These complexities are the traits that hinder direct security assessment from being applied onto those systems. A system is either secure or insecure but it is very unintelligent to easily label a system wholly as secure. A simple analogy to this concept is that a person is considered bad guy when he had committed crime even only once but he is not proven good even though he did good things many times.

So how do we go about assessing the security of a system? Luckily computer experts of generations ago already thought hard of these problems. They had gradually developed systematic and methodical way of implementing the assessment. Since a system is very complex we would never thought of every possible ways the security might be breached. We instead objectively identify those assets in the system that we

wish to secure and take certain measures to assure the security each of them. What security measures we had chosen and how do we assured of its implementation are actually the cornerstone of security assurance assessment. We give a certain level of trust to a system when we know that certain security measures have been implemented and assured of correct implementations being carried out for every measure. But how do we know about the security features and assurance of a system or product? There is no shortcut solution for this problem. All of those must be judged based on evidence and analysis. That means the systems must be subjected to systematic evaluation and certification. After this has been done then we should trust the system according to what have been verified. We will look at the evaluation issues further when we discuss the various type of security assurance evaluation.

Computer security experts normally refer systems with higher security assurance as trusted systems instead of secure systems. This also has been reflected in the widely known documents such as Orange Book[1]. There are various attributes that differentiate the secure against trusted security quality. Pfleeger [Pfleeger 1997] highlight these distinctions in the following table.

Table 1-1 Qualities of Security and Trustedness

Secure	Trusted
• *Either-or:* Something either is or is not secure	• *Graded:* There are degrees of "trustedness"
• Property of *presenter*	• Property of *receiver*
• *Asserted:* based on product characteristics	• *Judged:* based on evidence and analysis
• *Absolute:* not qualified as to how, where, when, or by whom used	• *Relative:* viewed in context of use
• A *goal*	• A *characteristic*

Now we have clearly understood that in order to have a trusted system or product; we need to state its security objectives or features; evaluate it to ensure that it really got the stated security features and also evaluate it to give us some assurance on the correct implementation of those security features. In the next section we will look how all of these related activities coordinated together into a systematic scheme and identify who are the parties involved and what are the roles played by end users in the whole coordinated security evaluation scheme for trusted products.

2. Security Assurance Evaluation

Since most of the end users are just want to have trusted systems and they themselves are not expert in the security area, who are going to evaluate a product so that it can be verify for trustedness? In most of the evaluation schemes these tasks are

[1] US Department of Defense Trusted Computer System Evaluation Criteria (TCSEC), commonly known as the Orange Book because of the color of its cover.

handled by independent third party security experts. The evaluation results produced by them are then later certified by an authorized certification body for quality control, consistency and oversight. There are many evaluation schemes established worldwide but the majority of them are already in the stage of being replaced by an international standard scheme known as Common Criteria. For further info on the scheme please visit the website at URL: http://www.commoncriteria.org.

3. The Need for Trusted Operating Systems

Why bother about trusted operating systems? What is wrong with existing operating systems? Actually there is no right or wrong in this issue. Everything boils down to the issue of appropriate tools for specific requirements. Trusted operating system does offer extra security features that are highly required when operating in certain environment and subjected to certain threats. Due to certain availability issues and perhaps higher cost, it is up to end users to weight out those factors against the perceived domain of threats when make up the decision to use or not. Following are the typical uses for trusted operating systems:

- servers and workstations for security, military and intelligence communities or any highly sensitive environments
- firewall, VPN and proxy
- application level gateway
- Certification Authority (CA) servers and workstations for Public Key Infrastructure (PKI)
- server appliance
- anything that need extra security features of trusted operating system

The demand that gave birth to the trusted operating system normally originated from military or security related agencies. Even though they are normally operating in closed or secure networks and establishment, they have never underestimated the threats coming from insiders. They are the communities that are always very cautious about every single aspect of security threats to their operations and organizations. Therefore the majority of trusted operating systems have been designed according to security models influenced by military world. For examples are level of security, perceived threats, operating environments and etc.

During the early days saw the only consumer for these products coming from security agencies. Higher price range, military biased security model and coupled with obscure deployment issues had further prevented the products from reaching a wider market. It seemed like the market for trusted operating system would doom to fail or never grow further. This was not a good sign for trusted operating system developers since they had invested millions and spent decades to come up with the products. Additionally these products also were subjected to evaluation (a process which could take years to complete) before being certified by certification body. Thanks to the Internet

which totally changed the landscape of threats to the general public and also security communities. Trusted operating system slowly gaining attention and market acceptance resulting from unwieldy attacks coming from the Internet [Olsen 1997].

Most of the servers that formed the backbone of Internet are UNIX[2] boxes. This can easily be understood since the main network protocol for Intenet which is TCP/IP was first developed on UNIX. Other factors as claimed by certain quarters are lean and mean structure of the operating system that contributed to the stability and security. Nevertheless there is a weakness in the operating system that really undermined the security of a system. In a UNIX system, there is a designated userid called *root*. It is also referred as superuser account because it got all the power to do almost anything on the system. The initial objective of having this account on the operating system is to provide a controlled channel for system administrators to do their administration tasks. But the first generation of UNIX was not designed with the priority for security. Instead the multi-user performance is the one. There is no provision in the design to confined the damage that could be done should the account accessible to abuser. Knowing this fact, most hackers have made the success of securing this account as the objective of their holy war. Trusted UNIX restrained superusers by eliminating them and dividing their powers into what it calls *administrative* privileges. Each power requires that the user be explicitly authorized to use it by a privilege authorization. All users must login as ordinary unprivileged users. They then can assume administrative privileges if they are authorized [Samalin 1997].

According to various worldwide surveys, highest number of computer security breaches are actually committed by insiders. Among the favorites are leaking out company trade secrets to competitors, denial of services and data manipulation. Typical organizations usually try to impose some control on the information access by using the concept of *Need-to-Know* basis which is adopted from military policy. Equipped only with common commercial or free OS, what can be done are actually very limited. Users normally have to login to access the system and once granted access, further access to the filesystems only possible if the said users or the group they are associated with are permitted to do so. Group access control normally divided according to functional units inside the organization such as accounting, marketing and so on. Once a user got access to a file, there is no mechanism to prevent him or her from changing the access control of the file to whatever access permission he preferred. This access concept is called Discretionary Access Control (DAC) because access is controlled at the discretionary of the owner. DAC is one of security functionality that must be supported in order to achieve security rating C2 of the TCSEC. Common OS such as Windows NT, Windows 2000 and most of the UNIX-based are rated at C2.

Any organization that would like to have the final say or mandatory control of information flow should look for additional security functionality available only to the OS rated at B1 and above of the TCSEC. It is referred as Mandatory Access Control (MAC). Unlike DAC, which allows users to specify, at their own discretion, who can and cannot

[2] UNIX means all flavors of UNIX-based OS such as Linux, BSD-based and others.

share their files, MAC puts all such access decisions under the control of the system. You can't give another user access to one of your files unless that user has the necessary clearance to the file. In typical systems, MAC is used in tandem with DAC whereby MAC dictates major sensitivity levels and DAC for finer granularity access control confined under MAC. Another security feature that related to MAC is Compartmented Mode Workstation (CMW). It is a security specification produced by US Defense Intelligence Agency. Whilst MAC can be seen as enforcing hierarchical security delineation or the sensitivity levels, CMW is used to enforce need-to-know policy of the subject matters or horizontal compartmentalization. Most of the trusted OS in the market are rated at B1 with CMW and sometimes got security features of B2/B3. Extra CMW security features that enticing the security communities are secure windowing and trusted labeling in windows, trusted networking, trusted-path and least-privilege capability [Adams 1996].

When designing secure or trusted applications, every component that constitutes the system must be verified as trusted as the applications. It is useless to build a very secure application when the underlying structure that it depends on is not trusted. In a trusted operating system the parts that enforced the security policy is called Trusted Computing Based (TCB). Even though a trusted application can devise its own TCB on top of non-trusted operating system, it would not be effective because its TCB still rely on the non-trusted part of the underlying operating system for accessing low level services. This has been reflected in the event of successful open hacking attack against a trusted application based on non-trusted operating system ([Delio 2001],[Rapoza 2001]).

Besides military or intelligence agencies, there are other organizations which require a total compartmentalization of information based on sensitivity levels, subject matters or other delineation criteria. But computer systems never operate in isolation. There are normally connected to various computer networks inside or outside the organization. For these organizations, they normally have multiple separate networks separated by air gap for different sensitivity levels or for different organizational entities. As a result, they always end up with multiple workstations on a person's table denoting different sensitivity levels. It is a concept of *risk avoidance*. Each of the workstation running in specific level is sometimes referred as operating in system-high mode. Some people classify this kind of setup as multiple security levels instead of multi-level security (MLS) system. The disadvantage of this arrangement is that there are always four to six physical networks need to be maintained at every place. The same goes for the number of workstations required for each network. We can simulate the notion of multiple networks logically using VLAN of Layer 2 or Layer 3 switching, or possibly using IPSEC for constructing VPN. Different segment can be identified using UTP cable of different color. For workstations, possible solutions are to use MLS-based system or the one that support multiple virtual machines [Murphy 2001]. All of these alternative solutions can be seen as manifesting the concept of *risk management*. The problem is that certain highly sensitive organizations could not accept the risk of information leakage if there are some mishaps in the switches, signal crosstalk, miswiring of connections or breakdown of IPSEC stacks.

In short, anything that cannot be seen physically is not trusted.

4. Trusted Operating Systems

In this section we will look at several implementations of trusted operating systems. A trusted OS is still a full featured non-trusted OS in terms of common functionalities and major components. What makes it special is the additional security features that it support and modified architecture to reflect those features. Since most of the security models that used for the construction of trusted OS were biased to the needs of military agencies, many security policies enforced into the design of trusted OS reflected from those models. These agencies also normally the ones that sponsored most of the research and development related to the trusted systems. These systems include Trusted Solaris, SCO CMW+, HP-UX BLS+/CMW, Digital MLS+ and IBM AIX CMW. But most of these operating systems already being upgraded to cater the needs of business enterprises as well. They might be referred with different product names.

Releasing the need to have the security features of trusted operating systems for everyday use in servers connected to Internet, a few open source and commercial development initiatives have sprung up recently to accommodate this need. Following summaries about the projects are excerpts taken from their websites:

- **TrustedBSD**. It provides a set of trusted extensions to the FreeBSD operating system, targeting the B1 evaluation criteria. This includes introduction of a number of features, including trust management, fine-grained system capabilities, access control lists, security event auditing, and mandatory access control [Watson 2000].
- **SELinux**. The limitations of traditional MAC have inhibited its adoption into mainstream operating systems. A flexible MAC architecture called Flask is developed to overcome the limitations of traditional MAC. The NSA has implemented this architecture in the Linux operating system, producing a Security-Enhanced Linux (SELinux) prototype, to make the technology available to wider community and to enable further research into secure operating systems [Loscocco 2001].
- **LOMAC**. It is a dynamically-loadable security module for Free UNIX kernels that uses Low Water-Mark Mandatory Access Control (MAC) to protect the integrity of processes and data from viruses, Trojan horses, malicious remote users, and compromised network server daemons. LOMAC is designed for compatibility and ease of use – to be a form of MAC typical users can live with [Fraser 2001].
- **HP Secure OS Software for Linux**. It provides comprehensive prevention, containment and detection with multiple layers of security. The software protects crucial server components, including the operating system and application layer, and it helps prevent unauthorized communication between

programs, networks and files.

- **NetTop**. It would turn each computer into a number of virtual PCs running on a Linux computer that would sit on each worker's desk. The security system would erect supposedly impenetrable, but virtual, walls between public data and more sensitive information on the same computer [Lemos 2001].
- **DTOS**. A program to encourage strong, flexible security controls in next generation operating systems. The first main task on the program was to develop a prototype secure microkernel by incorporating security mechanisms into the Mach microkernel. The second main part of the program was to perform research into assurance techniques for microkernel-based operating systems, with an emphasis on the practical use of formal mathematical methods [Spencer 1998].

5. Issues

There are a few reasons why certain organizations shy away from the trusted operating system such as B1+, CMW and MLS:

- many commercial or existing applications cannot be run on the system
- more expensive and difficult to use
- interoperability problems with non-MLS system
- always lag behind commercial offering in terms of features and technology
- limited availability and support worldwide

The purpose of using an operating system is to support the operation of various applications software. Without any application, the hardware with installed operating system is almost useless. In typical organizations, we can easily find server applications such as email server, web server, dns server, database server and so on. In a highly dynamic industry such as software development, developers usually unable to produce applications for all kind of operating systems existed. Instead a few that controlled the market are chosen. When the market for trusted systems is very small compare to the rest, developers just could not find the business case to develop on that platform. They normally developed for the non-trusted version of the same operating system. Furthermore the most of the trusted operating system are developed without complying to the standard operating system interface or perhaps there is no good standard at all. Without standard, it makes developers more difficult to justify for developing many versions of applications in already niche market.

Sometimes the inherently different design of the trusted operating system itself that makes the common applications broke down. For instance, certain applications developed for UNIX-based operating systems are expecting the normal functioning of the root account; but in trusted version of the same operating system, this root account is removed or behaves differently.

Research and design for the trusted operating systems normally requires more

resource to implement. For example it took Sun 12 years and USD50 million to develop Trusted Solaris 2.5 [Olsen 1997]. Considering the very niche market for trusted system, this fact is very hard for developers to accept. Therefore this added cost would be delegated to the consumer contributed to the higher cost of a typical trusted version of the same operating system.

Trusted systems sometimes need to be connected to the non-trusted system. Trusted systems with MAC and MLS features normally got interoperability problems when try to communicate with non-MLS one. For instance some problems may arise because of failure during the mediation of protocol between networking stacks, database records and authentication.

Despite tremendous research conducted in operating systems security, the capabilities of trusted operating systems a few years ago still are still below par. Certain trusted operating systems took between 1 to 3.5 hours just to boot up [Adams 1996], figures that never heard by typical PC users. But the situation is already improved nowadays. Most of the trusted systems just got a lag of between 6 to 9 months to their commercial siblings and the performance gap is not that wide. They achieved this by injecting secure components into the vanilla operating system. Pfeeger [Pfeeger 1997] argued that this approach could not be done to the trusted system rated at B2 and beyond.

One of the criteria for multinational organization when selecting vendors for IT procurement or support services is the availability of the items worldwide. Most of the trusted products are difficult to be obtained outside of US. This may be due to original contractors for producing the products are mostly from US government. Consequently certain features designed for trusted products are tailored towards the requirements of US government agencies or related to the military environment. Other probability is that other countries are not as serious as US in handling the digital sensitive information or perhaps bulk of their classified information is still not in digital form.

6. Suggestions

Considering the importance of guarding our information in the Internet age, we must take some proactive efforts to make the proliferation of trusted systems as ubiquitous as possible. We have seen how trusted systems did have some features to prevent some exploitation before it happened. Some suggestions on how to achieve this are as follows:

- establish internationally recognized standards for trusted system interfaces
- quickly produce trusted version of critical server software such as database, web, email, dns, application, middleware and so on
- extend the concepts of trustedness into the firmware and hardware domains
- regulate laws to make the use of trusted systems in certain environments as mandatory

It is the fact that software developers tends to produce software for the biggest

market. It is a chicken and egg problem. A market will not grow without having the applications first. In order to create significant market for trusted system, we must entice or reduce the barrier for software developers to write application software for trusted system. Therefore we must avoid or minimize the multi variation of flavors as what was happened to the UNIX market. We must make it easier for software developers to produce applications for trusted systems. We should establish standard interfaces or API for various trusted computing subsystems. For instance we should have standard interface for trusted operating system, data access for database, distributed object model, transaction, middleware, networking and many more.

As mentioned previously, a computer with trusted operating system but without applications is useless. We must have all critical trusted server applications ready before asking anyone to adopt it. Actually all of the suggestions above are inter-related to one another.

When we talked about a system, normally it is consisted of software and hardware. So it is incomplete if we just apply the concept of trustedness only to the software portion. Besides typical electronics components, most of the hardware also contains some elements of software which are never or seldom changed. They are normally stored in non-volatile RAM and known as firmware. So the construction of certain groups of hardware need to take into account some aspects of trusted features of other subsystems of which they might be deployed together. For example, a network device could incorporate the MLS feature into its core engine.

Sometimes business organizations would not do something until it is required by the laws. In this aspect, law can be considered as an effective vehicle to enforce the adoption of trusted systems. For example, the law should make the mandatory usage of trusted systems if an organization needs to hold some private information of the public such as credit card numbers.

7. Conclusion

We have presented the concept of trusted systems and highlighted some compelling reasons why and when we need to consider using trusted systems. But there are still a lot of issues need to be addressed before we can successfully adopt a holistic approach to trusted computing society. The Internet revolution really helped in bringing back the interest towards trusted systems into the information security arena.

With the rapid pace of development in information systems and Internet, sometimes we forgot to look into the underneath security of the infostructure. Coupled with the need to conduct business at blazing speed and mountains of backlog in software developments, the fragile security foundation in the computer systems laid unnoticed. But it is still not late. Even though there are monumental challenges ahead, with perseverance and cooperation from security communities, it would be achieved. Just like other efforts such as replacing IPv4 with IPv6 and converting all POTS into ISDN, they are massive

but still moving.

8. References

[Adams 1996] Adams, Charlotte. "Comparmented Mode Workstations still in demand for secure applications." Federal Computer Weekly. 23 Sep. 1996. URL: http://www.tcs-sec.com/media-reports/media-fed-comp-wkly-23sep96.html (22 Mar. 2002).

[Delio 2001] Delio, Michelle. "Hackers Win Security Challenge." Wired News. 23 Apr. 2001. URL: http://www.wired.com/news/print/0,1294,43234,00.html (22 Mar. 2002).

[Fraser 2001] Fraser, Timothy. " LOMAC – MAC You Can Live With." June 2001. URL: http://opensource.nailabs.com/lomac/ 22 Mar. 2002).

[Lemos 2001] Lemos, Robert. " NSA attempting to design crack-proof computer." 2 Feb. 2001. URL: http://news.zdnet.co.uk/story/0,,s2084134,00.html 22 Mar. 2002).

[Loscocco 2001] Loscocco, Peter. Smalley, Stephen. " Integrating Flexible Support for Security into the Linux Operating System." June 2001. URL: http://www.nsa.gov/selinux/freenix01-abs.html 22 Mar. 2002).

[Murphy 2001] Murphy, Dave. "NSA Closes Air Gaps." ITINFO. 2 Feb. 2001. URL: http://www.dgl.com/itinfo/2001/it010202.html (22 Mar. 2002).

[Olsen 1997] Olsen, Florence. "Sun finds Net creates users for secure OS." Government Computer News. 4 August 1997. URL: http://www.gcn.com/archives/gcn/1997/august4/cov2.htm (22 Mar. 2002).

[Pfleeger 1997] Pfleeger, Charles P. Security in Computing. New Jersey: Prentice-Hall Inc, 1997.

[Rapoza 2001] Rapoza, Jim. "Latest PitBull attack holds lessons for IT." eWeek. 30 April 2001.
URL: http://www.zdnet.com/filters/printerfriendly/0,6061,2713689-92,00.html (22 Mar. 2002).

[Samalin 1997] Samalin, Samuel. Secure UNIX. New York: McGraw-Hill, 1997.

[Spencer 1998] Spencer, Ray. "DTOS Program Summary." DTOS Home Page. 2 Mar. 1998. URL: http://www.sctc.com/randt/HTML/dtos.html (22 Mar. 2002).

[Watson 2000] Watson, Robert N. M. "Introduction to TrustedBSD." TrustedBSD Whitepapers. 3 Apr. 2000. URL: http://www.trustedbsd.org/documentation/whitepapers/introduction.html (22 Mar. 2002).

Last Updated: July 25th, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event