Operating System Security – A Short Note

[1,2]Mr. Kunal Abhishek, [2]Dr. E. George Dharma Prakash Raj
[1]Society for Electronic Transactions and Security (SETS), Chennai
[2]Bharathidasan University, Trichy
kunalabh@gmail.com, georgeprakashraj@yahoo.com

1. Introduction

An Operating System (OS) is viewed as a Reference Monitor (RM) or a Reference Validation Mechanism (RVM) that provides basic level security. In [1], Anderson reported three design requirements for a Reference Monitor or Operating System. He suggested that an OS or RM should be tamper proof that means OS programs are not alterable, OS should always be invoked and OS must be small enough for analysis and testing purposes so that completeness of which can be assured. These OS design requirements became the deriving principle of OS development. A wide range of operating systems follow Anderson's design principles in modern time. It was also observed in [2] that most of the attacks are imposed either on OS itself or on the programs running on the OS. The attacks on OS can be mitigated through formal verification to a great extent which prove the properties of OS code on various criteria like safeness, reliability, validity and completeness etc. Also, formal verification of OS is an intricate task which is feasible only when RVM or RM is small enough for analysis and testing within a reasonable time frame. Other way of attacking an OS is to attack the programs like device drivers running on top of it and subsequently inject malware through these programs interfacing with the OS. Thus, a malware can be injected in to the sensitive kernel code to make OS malfunction. However, virtualization technology is one of the solutions to counter both kinds of attack on OS as well as the programs running on top of the OS. The virtualization technology initializes hardware resources efficiently and is related to all the design requirements of an RM or RVM proposed by Anderson [1]. Additional advantage of using virtualization is to provide basic infrastructure which are used for cloud computing these days. Section 2, 3 and 4 gives a bit detail about these technologies to give countermeasures to OS attacks. Section 5 gives what needs to be trusted to trust a kernel. Section 6 gives a short discussion on what needs to be done to ensure OS security. Finally, section 7 concludes the discussion.

2. OS Verification

An OS is nothing but a bunch of compact programs and routines that interacts with hardware and manages resources to serve applications running on top of it. It is therefore important to verify each line of OS program code to ensure nothing is executing out of specifications. There are several methods to verify an OS among which three methods are popular. First, a Theorem Prover which ensures safety and reliability of the OS kernel. It can leverage the kernel behaviour in terms of state transitions of an abstract state machine. Theorem Prover can help to prove that the OS kernel does not allow illegal memory operation, unexpected halting or infinite looping causing deadlock like situations. Advantage of using a Theorem Prover method is that it can verify any arbitrary properties that can be input to the system. However, it is limited by the huge cost of verification due to manual construction. Second, the Source Code Modelling method checks for a model to be extracted from the source code given as input and exhaustive search of states are carried out to verify and guarantee the program properties. Advantage of this method is that no manual construction is made like the Theorem Prover method and therefore, it is a comparatively easier method to perform. But this method demands huge computational resources. Nucleus part of Verve OS which is verified and guaranteed type safety is the example of kernel verification through source code modelling. Third, use of Safe Programing Language guarantees OS program against strict type checking. Advantage of this method is that OS verification becomes automatic and also requirement of computation resources is low as compared to source code modelling method. But this method has some disadvantages too like verification of the OS is limited to basic properties only and it imposes high load on OS developers. Also, verification of OS with huge code base is also infeasible.

3. Access Control Mechanisms

Objective of Access Control Mechanisms is to ensure confidentiality, integrity and availability of the OS modules and is provided at the top of a Trusted Computing Base (TCB). A TCB consists of RM or RVM component. An access control mechanism essentially provides security policy modes, security policy description languages, security policy verification techniques and list of access control mechanisms. However, it is a challenge [2] to formulate access control mechanisms in cloud environment with virtualization as well as in embedded systems.

4. Virtualization Technology

Objective of virtualization is to give OS isolation for security. There are four main aspects in which virtualization can be enforced. First is Virtual Machine Monitoring (VMM) which can be performed through hypervisors. A hypervisor sits between virtual machine (VM) and hardware to performs resource management and scheduling of virtual machines. Use of hypervisor tightens access control over general purpose OS. Benefits of VMM with hypervisor include simultaneous operation of multiple virtual machines (VMs) which allows hardware resources to be utilized dynamically according to the operational situations. Hypervisors allows construction of observations and analysis devices which are not detectable by malwares and hence security of OS is ensured through virtualization technique. Virtualization implementation can be done through virtual machine inspection (VMI) which is a process carried out through hypervisors to allow monitoring, observations, defence and isolation of VMs. VMI is a very crucial component of virtualization technique as it detects and

prevents unauthorized access by VMs to alter the code on the hypervisor side and let hypervisor be able to monitor all aspects of the VM. A VMI is invoked to externally obtain the VM status for analysis of a snapshot of memory.

Second, virtualization can be enforced by main memory devices in which physical address used by VM are the artificial address virtualized by the hypervisors. Coordinating access to true physical addresses therefore requires twice address conversions causing double paging. This double paging can be performed in either hardware or software in case of VM technologies.

Third, virtualization can be implemented in Input-Output (I/o) devices through hardware using Input Output Memory Management Unit (IOMMU) or software using device drivers to capture and interpret the requests made by I/o devices. The IOMMU implements address re-mapping in hardware during direct access memory (DMA) allowing VMs to directly manipulate addresses of physical devices. Virtualization of I/o devices provides access control and integrity verification.

Fourth aspect of virtualization is verification of the completeness of VM through integrity verification of structural elements of VM through Trusted Platform Module (TPM). TPM is used for integrity verification of physical machines while virtual TPM (vTPM) is used for structural verification of multiple VMs running on top of a physical machine.

5. What Makes a Kernel Trustable [3]
It is feasible to build small systems that can be guaranteed never to operate out of specifications. Following OS interfacing programs need to be trusted to trust the kernel as they can propagate malware into kernel code to make it malfunction:

i. VM Encapsulation
Virtualization is supposed to be a cure for all ills relevant to security and therefore, the idea also comes to build a complete OS based on a hypervisor using VMs to encapsulate individual activities. A formally verified hypervisor like seL4 [3] can provide real security using virtualization and based on that a system i.e. operating system like CubeOS can be made.

ii. Web Browser
A web browser renders engine for each page in separate process inside an OS sandbox. The rendered processes can only access the system resources and communicate with each other via browser's kernel. Encapsulating security policies in separate module called browser's kernel is a security assurance that kernel would be safe. A typical web browser architecture [3] is shown in Figure 1 below to explain how a kernel does not allow any malicious web page to bye-pass security policies invoked by the kernel.

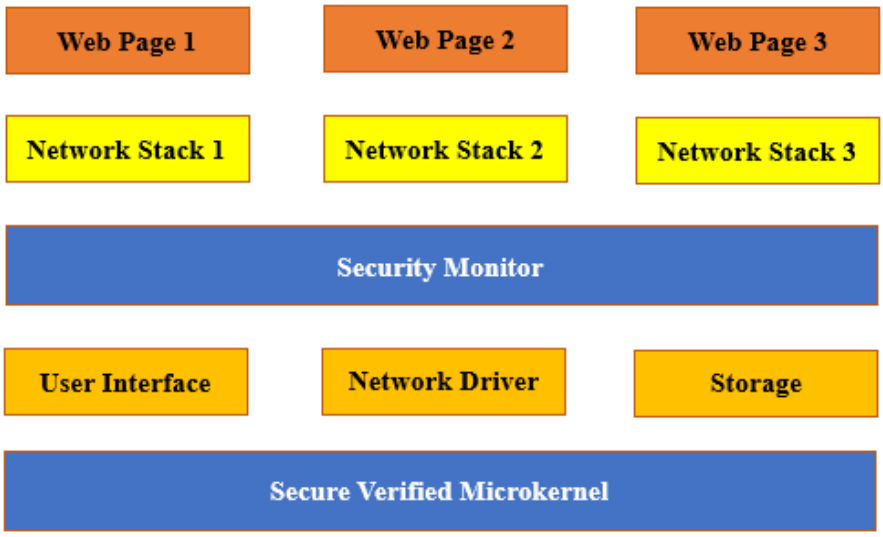

Figure 1: Secure Web Browser Architecture [3]

iii. Trusted Platform Module (TPM)
Trusted Computing Group has introduced the term TPM. A TPM enables secure boot, storage, authentication and remote attestation. Remote attestation ensures that system is running a well-defined software configuration and is achieved by accumulating hashes called measurements of software located in the system.

iv. Database
A database could pass atomicity, concurrency, integrity and durability (ACID) properties of transactions to ensure it would not harm the kernel programs.

6. How to Ensure OS Security?
The security of OS can be ensured through the following methods [4]:

i. By Proving Security
OS security can be proved through formal correctness. We need to specify the system i.e. OS as an Abstraction State Machine with completely defined input and output. Any variation in the expected output results violation in the OS integrity. A proof here identifies assumptions and ensures that the system is deployed under the right operating conditions.

ii. Safety
A Worst-Case Execution Time (WCET) Analysis is a powerful mechanism to ensure OS security through timing validations. WCET of an OS program code is the maximum time required to execute a given piece of code in a given application context (inputs, state) and on a given machine. The goal of WCET analysis is to derive upper bounds for the execution time of pieces of code [5]. WCET analysis is therefore performed to check reliability and functional correctness of the OS.

7. Conclusion
In this short note, we discussed about operating system design requirements in context of security. It was observed that an OS can be attacked either by attacking OS program code directly or through the application program interface to inject malware in to the OS. Some countermeasures like OS verification, access control mechanisms, virtualization techniques etc. were also discussed to prevent both kinds of attack on the OS. A brief note was also presented to arrive at what needs to be trusted to trust a kernel and what needs to be done to ensure OS security in general.

Acknowledgement
This note is an extract of cited research articles in the Reference section. We wish to thank the authors of [2], [3] and [4] for their commendable work which are reproduced in this article in the way to the best of our understanding.

Reference
[1] Anderson, J. P.: Computer Security Technology Planning Study, Technical Report Vols. I and II, USAF Electronic Systems Div., Bedford, Mass (1972)
[2] HASHIMOTO, Masaki. "A Survey of Security Research for Operating Systems."
[3] Heiser G, Ryzhyk L, Von Tessin M, et al. (2011) What if you could actually trust your kernel. In: 13th Workshop on Hot topics in Operating Systems (HotOS)
[4] G. Heiser, T. Murray, and G. Klein, “It’s time for trustworthy systems,” IEEE: Security & Privacy, vol. 2012, no. 2, Mar 2012
[5] https://ti.tuwien.ac.at/cps/teaching/courses/real-time-systems/slides/rts06_wcet_analysis-1.pdf

About the authors

Mr. Kunal Abhishek is a Scientist at SETS, Chennai with over 13 years of experience in design and development of Cryptographic and Network Security products/solutions. He also served as Software Engineer in Weapons & Electronic Systems Engineering Establishment (WESEE), an R&D unit of Indian Navy for 7 years. He was instrumental in framing "Digital Signature End Entity Rules, 2015" with inclusion of Elliptic Curve Cryptography (ECC) for Public Key Infrastructure (PKI) services under the IT Act of India. His research interest includes ECC based PKI and Secure Kernel Development. He holds an M.S. degree from BITS, Pilani and currently pursuing Ph.D. in Computer Science from Bharathidasan University, Trichy.

Dr. E. George Dharma Prakash Raj completed his Master Degree in Computer Science and Masters of Philosophy in Computer Science in the years 1990 and 1998. He has also completed his Doctorate in Computer Science in the year 2008. He has around twenty nine years of Academic experience and twenty one years of Research experience in the field of Computer Science. Currently he is working as an Asst. Professor in the School of Computer Science, Engineering and Applications at Bharathidasan University, Tiruchirappalli, India. Twelve of his PhD scholars have completed their PhD under his guideship. He is an Editorial Board Member, Reviewer and International Programme Committee Member in many International Journals and Conferences. He has published several papers in International Journals and Conferences related to Computer Science He has convened many National and International Conferences related to Computer Science. His Areas of Interest are Computer Networks, Big Data, Cloud Computing and Internet of Things.

Operating System – Security: Security refers to providing a protection system to computer system resources such as CPU, memory, disk, software programs and most importantly data/information stored in the computer system. If a computer program is run by an unauthorized user, then he/she may cause severe damage to computer or data stored in it. So a computer system must be protected against unauthorized access, malicious access to system memory, viruses, worms etc.
https://www.tutorialspoint.com/operating_system/os_security.htm