Proceedings of the 14th Colloquium for Information Systems Security Education
Baltimore Marriott Inner Harbor
Baltimore, Maryland June 7 - 9, 2010

Virtual Laboratories for Learning Real World Security - Operating Systems

Abstract – We present an extension of our virtual laboratory series with a new module that examines vulnerabilities in operating systems. These virtual laboratories aim to (i) model real world situations and bring together concepts from different knowledge areas, (ii) integrate theoretical knowledge and practical skills, and (iii) give students the opportunity to execute the laboratory on virtually any system and experiment anytime, anywhere in a secure, easily accessible environment.

This new laboratory demonstrates how a kernel rootkit can be used to compromise a system. In a series of activities including installing, certifying, and working with a vulnerable system, code is evaluated and discussed in the context of the theoretical knowledge acquired in our core courses in digital forensics, network and software security and operating systems. Students are given the ability to play different roles and analyze security issues from the perspective of an attacker, systems manager, and forensic analyst. The implementation is based on Linux and the module is integrated in the Blackboard Vista course management system. The laboratory includes an online learning module that takes advantage of many interactive and self-learning tools such as video collaboration technologies, online student-faculty and student-student discussions, rich multimedia content, step-by-step video demonstrations of selected laboratory activities, self-evaluation and graded assessments. The exercises and assessments are designed mainly for the courses in operating systems, networks, and digital forensics, but have a range of activities that are suitable for a variety of programs and concentrations.

Index terms – information security curriculum, operating system, digital forensics, virtual environment, online and blended learning.

I. Introduction

Operating systems are at the core of the global information infrastructure, from world-wide computer networks to mobile phones and other popular gadgets. The greater flexibility and wide distribution of today's operating systems leads to increased vulnerability. Thus, an understanding of operating system security is no longer limited to the system manager, but becomes a general concern.

A basic design principle of The University's curriculum in information assurance is to enable students to draw upon knowledge from different fields—e.g. cryptography, networking, economic incentives theories—and successfully apply diverse concepts and methods in problem solving. This can be effectively achieved through laboratories that model real world situations and allow students to play different roles, exploring opposing or complementary viewpoints, as will be the case in typical organizational structures. The laboratory experience has the added benefit of naturally integrating theoretical and practical skills. Towards this goal we started developing virtual laboratories [1] that can be run on any system and be available anytime and anywhere. We continue to stress that the lack of hands-on experience and/or the failure to explain clearly and in detail how theory underlies practical solutions may leave students with fragmented knowledge and insufficient practical skills. Our laboratory modules integrate knowledge from several security courses and include configuring, and working with widely used commercial and open source applications in a virtual environment. For each laboratory activity we trace the relevant theoretical concepts and discuss their implications for securing the system. The choice of working with real-life role-based environments instead of developing targeted laboratory exercises for specific concepts is deliberate and important— students acquire practical knowledge of modern information security technologies and skills that are readily applicable in the workplace. They are better prepared to explain security problems, raise security awareness and build more secure infrastructures.

In few fields is the contrast between theory and practice as stark and important to reconcile as it is in information security. This is why our first laboratory module focused on relating encryption algorithms with Public Key Infrastructure in a scenario that is typical for many industry applications and includes installing an application server, establishing secure communication between client and server, generating public and private key pairs, and requesting and obtaining certificates from a Certificate Authority, [1]. This paper takes our approach in a new direction: it looks into the heart of today's information infrastructure that extends from global computer networks systems to billions of ubiquitous gadgets – security of the Operating System (OS). The core of our security curriculum includes courses in operating systems, forensics, and network and software security. Our new module, while touching on theoretical concepts from all three courses, focuses mostly on the

operating systems course, bringing into it laboratory activities in a virtual environment. Throughout the labs we emphasize three aspects: (i) the basic theoretical concepts and methods to achieve confidentiality, integrity and availability; (ii) the technical implementation; (iii) the human factors and typical organizational roles such as the systems manager and forensic analyst, and how they interact with an attacker.

It is of critical importance to build into the curriculum elements of practical cyber education, not just from readings and lectures, but through practical experience. This gives students an insight into the thought process and intentions of potential adversaries. Teaching students these concepts will ensure they keep them in mind when moving to the real world. In the field of information security, threats move from the theoretical to the real in mere moments. As soon as a vulnerability is discovered, there are individuals attempting to find ways to exploit it. Keeping up with these ongoing threats is vital not just for industry but for government as well.

Another important skill students need to be exposed to is that of researching solutions to unknown problems. It is very rare in the real world to be presented with a problem as well as all the information required to solve it. Students need to learn to navigate through the expansive resources available to them in order to research solutions and methods which may not have been covered in class materials. This is not to say that students should not ask questions of their professors and classmates, but they should be capable of doing research on their own to broaden their knowledge. Our laboratory module aims to develop a healthy curiosity such that students desire to learn more.

The last decade saw a strong growth in distance and blended delivery as well as extensive use of educational technologies in the traditional classroom. Our programs are delivered in all three of these formats. This module was developed as a laboratory that can be used in a traditional classroom, in an online course, or as a combination of classroom and online activities. We have developed comprehensive online materials to provide background knowledge, references, and descriptions of laboratory activities. We have made access to the laboratory and online content available through the Blackboard Vista course management system – a platform prevalently used in our online and blended programs, but also available in our technology-enhanced classroom courses. The learning system helps to provide a blended experience to students by easily integrating collaborative discussions, video tutorials and direct links to complementary materials.

We have also build upon our existing use of virtual laboratories and created a ready-to-run virtual machine (VM) containing all elements of the lab which students can run virtually anywhere and on any system. The virtual environment contains all tools and documentation needed to successfully complete the laboratory and can be distributed as a DVD or through download; to use it students install a freely available VMware player that works on most popular operating systems. Using a virtual machine saves the student from wasting time obtaining, installing and configuring the necessary tools, and provides a safe environment for experimentation without a risk of compromising their personal computers.

II. Operating System Security and Digital Forensics

The background knowledge for the new laboratory module draws mainly from our courses in operating systems security, digital forensics, and networking. Rootkits attack the security of the operating system kernel, therefore a basic understanding of operating system structure and function is required for completing the laboratory activities. Rootkits are also a type of malware and therefore can be analyzed with current digital forensics techniques. In a highly interconnected world with practically no stand-alone systems left, network knowledge is indispensable for understanding how a system can be compromised and determining effective counter measures. The rootkit module will be included in all our operating systems, forensics, and networking courses, as well as being made available to related courses.

Operating systems are at the core of an information infrastructure. It stands to reason that they would be an ideal attack vector for compromising a system. IT professionals need to be keenly aware of this vulnerability; however, many do not truly understand the workings of an operating system kernel and assume it is obscure enough that it would be quite difficult to compromise. Since in-depth operating system security is not a commonly covered topic in the classroom, students may not be aware of the implications of rootkit infection. In addition, traditional concepts for information security based on confidentiality, integrity, and availability analysis are sometimes difficult to apply to OS Security. We need to educate students through practical exercises of the unique challenges rootkits pose, and increase awareness to this particular security threat. This laboratory intends to begin this education and demonstrate just how easy it is to compromise a typical system.

In the laboratory we stress that fundamental security principles must be followed to ensure the security of the operating system itself. After successfully completing the

lab, students will gain a good understanding of topics such as securing a segment of code, weakness of the "security through obscurity" approach, limitations of encryption in protecting confidentiality, achieving tamper-proof integrity of key kernel components, preventing unauthorized system modifications, eliminating threats to non-repudiation attributable to rootkits, etc. The laboratory is designed to have students thinking about these topics while completing the laboratory exercises, further preparing them for situations they can and will encounter in the real world.

Modern operating system kernels have built-in security mechanisms that make it more difficult for malicious users to modify key parts of the operating system. Industrious attackers have, however, found ways to work around or disable these mechanisms, as we demonstrate in the lab. With the requisite knowledge (which is freely available from a number of sources online), a motivated individual can work around most any of these types of mechanisms. With the attacker utilizing a low (assembly language) programming level it is very hard to prevent these types of attacks. This is especially a concern with open source operating systems where all the source code is available to anyone who cares to download it. This is not to suggest that other operating systems are less vulnerable. Multiple times segments of code from commercial non-open source operating systems (i.e. Microsoft Windows) have been made available. Additionally, it is possible to reverse engineer compiled programs to low level readable code which can closely resemble the source code. Having the source code or disassembled instructions makes it much easier to understand and bypass security mechanisms. In our laboratory, students work with the source code of a rootkit to bypass kernel security mechanisms and compromise a target machine.

Digital forensic analysis is another focus of this laboratory which reinforces techniques that can be used after the rootkit has been detected. A rootkit is a type of malware just like a virus or a worm – in fact, a rootkit in a sense is an evolution of traditional malware. Rootkits frequently come packaged into viruses and worms. They are a convenient way for malicious users to compromise a system. In this laboratory students use malware forensic analysis techniques to determine the scope of impact of a rootkit infection. Students are encouraged to consider questions related to establishing the source of an attack as well as attack vectors, attack date and time, which vulnerabilities were exploited, rootkit capabilities, rootkit identification, what information was lost or compromised, etc. Tools are included in our laboratory VM image to accomplish some of these tasks, but others will require the student to scour log files or read through the source code of the rootkit for clues. These are all valuable skills to have in the real world when analyzing an infected system.

We would be remiss to talk about any topic within information security without some mention of security protocols. Security protocols come into play in every area of computer science. The problem is that the kernel runs at a level which supersedes that of most security protocols take effect. Rootkits run so close to the hardware that they can in effect bypass most mechanisms put into place in code. Traditional security protocols rely heavily on authorization. Rootkits completely bypass this protection mechanism by accessing the system at such a low level that it is below the implementation of any security protocols.

III. Laboratory Background

Implementing a rootkit lends itself well to a laboratory exercise. Several different aspects of information security can be explored throughout the execution of the lab. Digital forensics comes into play when attempting to detect the rootkit and determine what it has done to compromise the system, as well as gaining an understanding of how the rootkit persists.

While rootkits can be implemented on virtually any type of system, we have chosen to focus on Linux. We use a CentOS distribution based on Red Hat source code version 4.1.2-44. It is running kernel version 2.6.18-128.4.1.el5. We chose Linux for simplicity in use and distribution (it is free and not encumbered with licensing issues). In general, Linux systems work best due to the fact that they are freely distributable and students can retain the virtual image of the system after the laboratory is complete to further explore the topics discussed in the lab, expanding on them without thinking about licensing implications. It is important to note that in our laboratory we make no assumption that the student has any particular knowledge of Linux. General computing knowledge is all that is required. All steps are thoroughly explained and all tools and instructions needed are made conveniently accessible.

The exercises are built around compromising an operating system kernel. Students learn that the kernel is the central component of the operating system which can be thought of as the bridge between applications and the actual data processing done at the hardware level. Key concepts and ideas they are exposed to from a security perspective are that the operating system kernel operates at the highest privilege level of the system (Ring 0 in Figure 1 below) and the consequent high risk of malware operating from this level. The laboratory illustrates the lack of protection and freedom of code execution at this level which is very appealing to individuals looking to gain control of a computer. We show how difficult it is to protect systems and detect intrusions at this level.

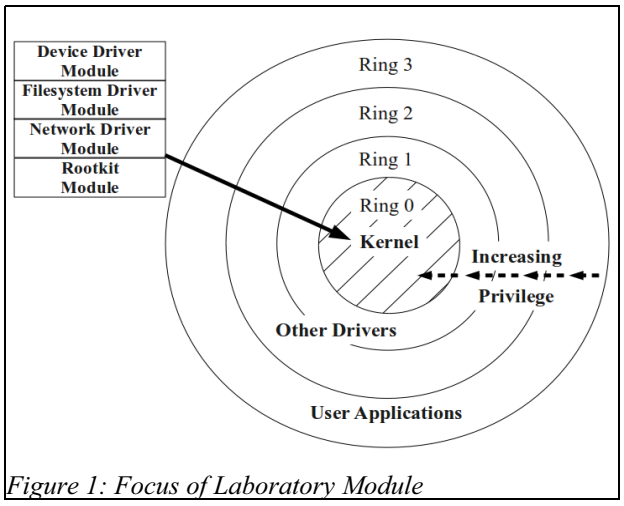

Figure 1: Focus of Laboratory Module

Students also learn about kernel modules, which are segments of code that can be loaded and unloaded into the kernel upon demand. They learn that a rootkit is actually a module which is loaded into the kernel just like any other legitimate module such as a device or network driver. They are able to visualize how a rootkit could easily seize control of the operating system and even obscure the fact that the system has been compromised. Figure 1 points out the focus of this laboratory as Ring 0, running at the highest privilege level where kernel modules are loaded.

This laboratory has been designed to allow students to play different roles as they relate to the system: e.g. attacker, systems manager or forensic analyst. Depending on the size of the class, these roles can be interchangeably played by one or several students, and large classes can form student groups that work on several parallel environments. The three roles mentioned above give students an insight into the continuous struggle that exists between those who intend to compromise systems and those whose job it is to defend them.

IV. Laboratory activities

In this laboratory students experiment with compiling and inserting a kernel mode rootkit which hijacks various system calls and modifies their functionality. These are typical tasks for any rootkit which an attacker might use to compromise a system. Understanding the functionality of the rootkit code, its detection and mitigation is essential for any security professional. This laboratory provides important practical knowledge of operating system rootkits which supports that understanding. Additionally students will see how trivial it is to write effective rootkits, and how difficult it is to detect them.

The laboratory provides source code for a kernel rootkit which can be used to hijack certain system calls. The code is documented so that even a novice coder is able to follow it to the degree necessary to complete the lab. The students then compile the code and examine the output. Instructions are given on how to insert the module and then confirm that the module has been successfully loaded. A few commands are tested to ensure that the module is working as designed.

The laboratory focuses on evolving learning through exercises, questions, and optional additional activities for the students. It contains two exercises that involve hijacking system calls, adding persistence to the rootkit and performing tasks to detect the rootkit. The last exercise contains several questions requiring the student to demonstrate sufficient understanding of the processes.

The learning objectives as listed in the laboratory are:

- Learn about OS kernel; develop hands on skills to work with OS internals.
- Gain an understanding and appreciation for kernel security concepts.
- Learn and practice digital forensics techniques to analyze a system infected with a kernel rootkit.
- Gain practical knowledge of security mechanisms and their proper use within a Linux kernel.

A. Exercises

The first “role” we discuss is that of the attacker. It should be noted that exercises for the second and third roles (systems manager and forensic analyst) both rely on the tasks of the first role (attacker) being completed successfully. These include compiling the code on a “target system”, inserting a rootkit module, testing its functionality and modifying the system such that the rootkit persists after a reboot. It is important for students to go through the exercises of compiling the code and ensuring persistence of the rootkit. This is what a real attacker would do in order to ensure their rootkit is

successfully planted into the targeted system. It also provides valuable clues for completing the forensics portion of the lab. While completing this section of the laboratory students are encouraged to look over the code and read the comments to develop a detailed understanding of interactions between operating system components.

The VM contains two different rootkits: The first corrupts the Open system call (one of the more common and easier calls to hijack), and the second corrupts the Write system call. Throughout the laboratory some values are required to be recorded to ensure proper execution of exercise steps. Below (Figure 2) is a screenshot of a portion of the laboratory being completed. It shows a request for a directory listing of the compiled rootkit module, and request for a list of all open type system calls. Next, a module is inserted into the kernel using the standard *insmod* command. The module reports successful insertion. The user again requests a list of all open type system calls and notes that the rootkit has generated a new one. Lastly, the rootkit is tested for functionality. Indeed students see in the command output that the rootkit is intercepting open system call events.

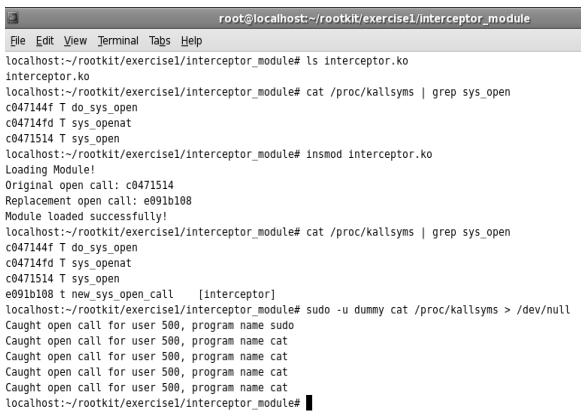

Figure 2. Laboratory fragment demonstrating a compromised system

§

Figure 3 is an example of graphical illustrations included in the online module. It shows modifications made by the rootkit, and how they relate to other OS components. Illustrations such as these help students to follow the laboratory process and cross-reference laboratory exercises with material learned in the theoretical portion of the curriculum.

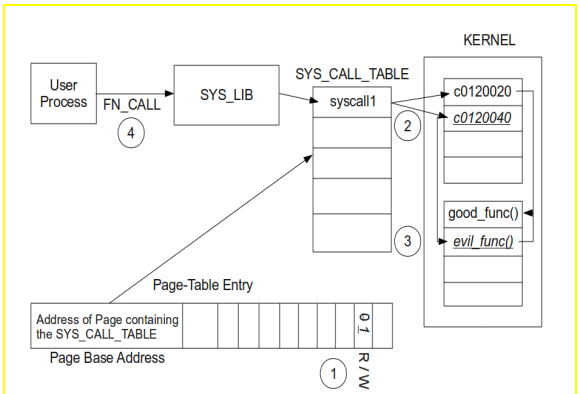

Figure 3. Sequence of Events

The second role students take is the role of a system manager on the targeted system. As system managers, students walk through detection, identification and removal of the rootkit. A virus scanner has been included into the VM image to demonstrate that most virus scanners do not detect rootkits. Common utilities to detect rootkits have also been included into the system image for students to use in the detection process. Students learn limitations of virus scanning tools and the importance of taking additional steps to protect against rootkit infections. The included rootkit will only be detected by "suspicion" like most tailored rootkits. The module will then be identified and successfully removed by the student. Steps will also be covered to reverse the process of persistence added in the attacker role. Fundamental information security practices for system managers, including the principle of *defense in depth*, are stressed. These experiences ensure that when students reach the real world they can work from the beginning with security in mind.

The third role the students take is the role of a forensic analyst. Forensic principles are followed to isolate the system, determine how and when the infection may have occurred by examining log files, determine the extent and capabilities of the infection by locating and reading the source code, and determine whether the rootkit can be safely disabled or removed. Adherence to the forensic process and proper documentation is stressed while completing laboratory activities. Appropriate forms are provided for the student to complete keeping the activities as close to real world as possible.

B. Assessments

A number of self-assessment and graded questions are included to enforce understanding and ensure students have thought through the lab. Various levels of questions are included, allowing faculty to select the proper set of questions to tune the laboratory to the curriculum and

course specifics. A solutions manual has also been developed to cross-reference questions with learning objectives. The assessments can easily be implemented in a course management system for consistent experience and ease of access for all students in online, classroom or blended settings, as well as to help faculty with grading. A number of questions go beyond what is covered in the exercises, requiring students to do some research on their own.

C. Advanced Exercises

We believe it is important to encourage students to delve further into topics which generate interest for them. To give them a place to start we have included several optional advanced exercises. These can also be used within the course for extra credit or as projects. The exercises include extending or improving upon the included code, as well as implementing new hijacked system calls. Students are encouraged to work in groups on these exercises and utilize online discussions available to them through the online course management system.

V. Conclusion and Future Work

We have presented a new laboratory module that follows an end-to-end security process pattern while raising awareness of security in operating systems. Our approach binds the theory of operating system security to practical knowledge about widely used systems and teaches the necessary skills for working with them. All laboratory activities are conducted in a virtual network environment that is secure, reduces costs, allows students to easily assume different roles—attacker, systems manager, forensic analyst—and gain a keen appreciation of the different perspectives.

We intend to continue developing similar real world laboratory modules to address complex security problems in the context of real world business scenarios using commercial and open source software, and integrate knowledge that is taught in courses across the security curriculum with valuable practical skills. In the process we will take advantage of the latest enabling technologies, including virtualization, as well as new pedagogical trends. Our laboratory modules lend themselves well to traditional classroom teaching, as well as to online and blended program formats. To support active online and blended learning, we incorporate a variety of approaches to manage course dialog and structure, and maintain focus on learning objectives, including the use of advanced course management systems, multimedia and video collaboration technologies. We always involve our graduate students in the process of developing and testing the laboratory modules, as they provide valuable feedback. Additionally they benefit from this work as it is a great learning experience for them.

This laboratory module will be used in a number of our courses and we hope students and faculty are as enthusiastic as they were with our previous module.

VI. Acknowledgements

We would like to acknowledge the help of all those who have helped make this effort possible, arranging and formatting our laboratory materials, helping us gather the materials for the paper, editing various drafts of the paper or feeding us tootsie rolls, your efforts are much appreciated. Special thanks go to our instructional designers, Rosemary Antonucci and Arpic Patel, our resident infosecurity hobbyist Kyle Brogle, other members of our MET Research team, MET Computer Science department, and MET IT group.

VII. References

[1] Zlateva, T., Burstein, L., Temkin, A., MacNeil, A., Chitkushev, L. Virtual Laboratories for Learning Real World Security, Proceedings of the 12th Colloquium for Information Systems Security Education, Dallas, TX, June 2008.

[2] Zlateva, T, V. Kanabar, A. Temkin, L. Chitkushev, S. Kalathur. Integrated Curricula for Computer and Network Security Education, Proceedings of the Colloquium for Information Systems Security Education, Society for Advancing Information Assurance and Infrastructure Protection, Washington, D.C., June 2003.

[3] Irvine, Cynthia E., Chin, Shu-Kai, and Frinke, Deborah. "Integrating Security into the Curriculum", IEEE Computer, Vol 31, No 12, pp.25-30, 1998.

[4] Dodge, Ron, Dan Ragsdale**: "**Technology Education at the US Military Academy", IEEE Security and Privacy, vol. 3, issue 2, pp. 49 – 53, March 2005.

[5] Bishop, M. "Teaching Context in Information Security," Journal on Educational Resources in Computing 6(3) article #3, September 2006.

[6] Ross J. Anderson: Security Engineering: a Guide to Building Dependable Distributed Systems. 3rd ed, Wiley Computer Publishing, 2008.