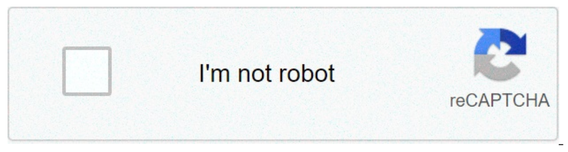

Continue

Hacking the art of exploitation 2nd edition pdf

Ed Schaefer Ed Schaefer reviews the newest edition of Hacking: The Art of Exploitation. Members of the Hacking world are known by the color of their hat – white for good, black for bad, and gray for those who aren't too sure. By writing Hacking: The Art of Exploitation, Jon Erickson proves his hat color is "mother of pearl." Don't let the title mislead you: Erickson isn't exploiting or vandalizing – he's instructing. In 2004, I reviewed the book's first edition. In my reviews, I typically like to compare the differences between editions. Erickson beat me to it. At the publisher's web site, you can compare the first and second editions of the book; view excerpts from the Exploitation, Networking, and Countermeasures chapters; and download the book's source. Erickson also bundles the source in a CD included with the book, but more on that later. Expanded Concepts Introduction In my first review, I recommended this book for the programming chapter alone. I can no longer do that because the programming chapter is now an "Expanded introduction to fundamental programming concepts for beginners." But it's like no introduction I've ever seen. In one chapter, Erickson takes us from basic Control Structures to Function Pointers. Think of it as Kernighan and Ritchie in 100 pages. Erickson covers other introductory topics in a hurry, such as his network sockets description in the Networking section (Chapter 4), and his "Crash Course in Signals" in the Countermeasures section (Chapter 6). It's not that I don't like the author's introductions – I do. I just want to warn you that the introductions might be above the true beginner's head. This book is code intensive and if you don't have a programming background – preferably in Linux "C" – then this book may be of limited value. If you aren't into hacking Linux, or at least wanting to learn, then this book just might gather dust on your book shelf. Should You Buy the Book? Because the programming chapter is now an introduction, I now recommend this book for the Exploitations chapter alone. This chapter covers buffer and function overflows and the format string vulnerability. Buy the book and discover why strings should be formated like this: printf("%s", text); and never like this: printf(text); What's on the CD? For readers with no access to a Linux box, Erickson bundles his source with a bootable Ubuntu Linux Live CD. The Live CD requires "an x86-based PC with at least 64MB of system memory and a BIOS that is configured to boot from a CD-ROM." I successfully booted the Live CD with both an IBM T43 laptop and a HP dv9000t laptop. Jon Erickson Paperback, 488 pages No Starch Press, January, 2008 ISBN-10: 1-59327-144-1 ISBN-13: 978-159327-144-2 US$ 32.47 This article has multiple issues. Please help improve it or discuss these issues on the talk page. (Learn how and when to remove these template messages) This article needs additional citations for verification. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.Find sources: "Hacking: The Art of Exploitation" – news · newspapers · books · scholar · JSTOR (July 2018) (Learn how and when to remove this template message) This article includes a list of general references, but it remains largely unverified because it lacks sufficient corresponding inline citations. Please help to improve this article by introducing more precise citations. (July 2018) (Learn how and when to remove this template message) This article contains content that is written like an advertisement. Please help improve it by removing promotional content and inappropriate external links, and by adding encyclopedic content written from a neutral point of view. (January 2016) (Learn how and when to remove this template message) The topic of this article may not meet Wikipedia's notability guideline for books. Please help to demonstrate the notability of the topic by citing reliable secondary sources that are independent of the topic and provide significant coverage of it beyond a mere trivial mention. If notability cannot be shown, the article is likely to be merged, redirected, or deleted.Find sources: "Hacking: The Art of Exploitation" – news · newspapers · books · scholar · JSTOR (January 2016) (Learn how and when to remove this template message) (Learn how and when to remove this template message) Hacking: The Art of Exploitation (ISBN 1-59327-007-0) is a book by Jon "Smibbs" Erickson about computer security and network security.[1][2] It was published by No Starch Press in 2003, with a second edition in 2008. All of the examples in the book were developed, compiled, and tested on Gentoo Linux. Author and background information Jon Erickson is a computer security expert, with a background in computer science. As of 2011, he worked as a vulnerability researcher and computer security specialist in California. A bootable CD is included with the book which provides a Linux-based programming and debugging environment for the users. Content of 1st edition The content of Exploiting moves between programming, networking, and cryptography. The book does not use any notable measure of real-world examples; discussions rarely bring up specific worms and exploits. Programming The computer programming portion of Hacking takes up over half of the book. This section goes into the development, design, construction, and testing of exploit code, and thus involves some basic assembly programming. The demonstrated attacks range from simple buffer overflows on the stack to complex techniques involving overwriting the Global Offset Table. While Erickson discusses countermeasures such as a non-executable stack and how to evade them with return-to-libc attacks, he does not dive into deeper matters without known guaranteed exploits such as address space layout randomization. The book also does not cover the Openwall, GrSecurity, and PaX projects, or kernel exploits. Networking The networking segment of Hacking explains the basics of the OSI model and basic networking concepts, including packet sniffing, connection hijacking, denial of service and port scanning. Cryptology The cryptology section of Hacking covers basic information theory, in addition to symmetric and asymmetric encryption. It winds out in cracking WEP utilizing the Fluhrer, Mantin, and Shamir attack. Besides the basics, including man-in-the-middle attacks, dictionary attacks, and the use of John the Ripper; Hacking discusses quantum key distribution, Lov Grover's Quantum Search Algorithm, and Peter Shor's Quantum Factoring Algorithm for breaking RSA encryption using a very large quantum computer. Other Details The front cover of Hacking is the complete cycle, from reverse engineering to carrying out the attack, of developing an exploit for a program that dies of a buffer overflow over long command line arguments. Content 2nd edition Hacking: The Art of Exploitation Second Edition AuthorJon EricksonCountryUnited States (Original)LanguageEnglish (Second Edition)SeriesSecond EditionGenreComputer SciencePublisherNo Starch PressPublication dateFebruary 2008Media typePrint PaperbackPages488ISBN978-1593271442 The introduction of the book states that hacking should only be done within the confines of the law, and only for productive reasons. 0x200 Programming In the programming chapter of this book, different types of programs are described and shown via examples. The chapter covers control structures and other basic aspects of programming. The live CD provides an environment so that the reader can not only follow along with the examples in the book but do some programming themselves. 0x300 Exploitation Exploitation is taking the computer's code or set of rules and changing them so the computer does what you want it to do. Finding ways or holes in the system to change is an important part of exploitation. This chapter covers exploit techniques such as memory corruption, Buffer overflows and format strings, especially using Perl and Bash shellcode. 0x400 Networking The OSI Model. OSI Model In communication among computers through networking the OSI Model is used. The OSI Model is a model that provides the standards that computers use to communicate. There are seven layers in the OSI Model and they are Physical layer, Data-Link layer, Network layer, Transport layer, Session layer, Presentation layer, and Application layer. Each packet that a computer sends out to another computer must go through each layer of the OSI Model. Sockets The standard way to communicate on a network with the use of computer operating systems is a socket. A socket is used by a programmer to create a way to send and receive data using the layers of the OSI. There are two types of sockets: a Stream Socket and a Datagram Socket. Stream Sockets use Transmission Control Protocol (TCP). A Datagram Socket uses User Datagram Protocol (UDP). Peeling Back the Layers 'Peeling back the layers' describes how the OSI layers actually work. The OSI Model is described in great detail with some images in the book that make it easy to understand. Network Sniffing Switched and unswitched networks exist in networking. A switched network uses devices called switches that analyze and tell the packets travelling on the network where their endpoint is. An unswitched network is a free flow of packets without them being stopped and analyzed. Sniffing refers to using a program that allows you to see packets on the network and where they are going. Denial of Service A denial of service attack is an attempt to make a computer resource unavailable to its intended users. This means that the denial of service attack sends a large quantity of communication requests to an intended resource in order to overflow the resource so that it becomes unusable for a certain period of time. These types of attacks are usually directed at routers or firewalls in order to shut them down to gain access to other computers on the network. A router is very susceptible to these types of attacks but a firewall can usually handle the attack and is unaffected. A distributed denial of service attack is when communication requests come from multiple computers, greatly increasing the number of requests over a regular denial of service attack. Some other types of DOS attacks include Ping of Death, Teardrop, Ping Flooding, and Amplification attacks. TCP/IP Hijacking TCP/IP Hijacking is another way that uses spoofed packets to take over a connection between the victim and a host machine. This technique is mainly used to collect passwords when a host machine uses a password to be connected to. When this type of attack takes place the victim and the attacker must be on the same network. Another form of TCP/IP hijacking is RST hijacking, which involves injecting a fake reset packet. Port Scanning Port scanning is simply a way to figure out which ports are accepting and listening to connections. The hacker would just use a program that lets him know which ports are open by scanning all the ports on a network and trying to open them. There are many other type of scans, such as SYN, Idle, FIN, X-Mas, and Null scans. Reach Out and Hack Someone This part is about finding vulnerabilities in the typecasting of the network. Using a debugger to go through lines of code which are used for network protocols is the most efficient way to accomplish this. 0x500 Shellcode Shellcode is used in the exploitation of code. It is like the hacker's own code that he wants to run when he gains control over a program. Usually a hacker will find an exploit in a programs code and be able to insert some of his own code (shellcode) where he found the exploit. Assembly vs. C Assembly differs from C because assembly is a low-level programming language, and when processed can communicate directly with the processor. When using C, which is a high-level programming language, the code must be compiled and sent to the kernel by making a system call, and then making a call to the processor. In other words, it is almost like taking the system calling to the kernel out of the picture when using assembly.[citation needed] The Path to shellcode This part of shellcode is about how to inject a program with shellcode so it can be taken over. There are many examples of code in the book and ways to accomplish this task. Self-spawning shellcode Spawning shellcode is code that will be enabled when an exploit is found. It is shellcode that will be able to be run when a vulnerability is found in the program. The best way to accomplish this is shown in the book and by making sure the code is very small. Port-binding shellcode This type of shellcode attaches itself to a network port. Once bound to a port it will listen for a TCP connection. After it finds the TCP connection there is a lot more programming involved and is shown vividly in the book. Connect-back shellcode This type of shellcode is mainly used when getting around firewalls. Most firewalls are going to block port-binding shellcode from working because they are set up to only allow known services through the active ports. Connect-back shellcode initiates the connection back to the hacker's IP address so it will be coming out from the firewall instead of going into it. Once again the code in the book depicts connect-back with the use of shellcode and ways to accomplish this. 0x600 Countermeasures This part of the book is about having defenses and intrusion prevention systems to stop known hacking exploits. Countermeasures That Detect An administrator of the network has to be aware of when an exploit may be occurring. Using certain tools like reading logs or packet sniffing on the network are a few ways to catch exploits when they occur. System Daemons A System Daemon is a server program on a Unix system which receives and accepts incoming connections. A daemon is a program which runs in the background and detaches from controlling the terminal in a certain way. At this point in the book there is some code shown on how to run a daemon program. Signals are also used in a Unix-based environment to make operating system calls. When a signal is type in the terminal it will immediately send an interrupt message to complete the task of whatever the signal was which was typed. The uses of signals are displayed in some coding examples in the book. Tools of the Trade A hacker has a certain set of tools that he needs to help him when exploiting. An exploit script is a tool in which uses already written exploit code to find holes in the system or program. Using exploit scripts is easy for even a non-hacker to use because the code is already written in it. A couple exams of some exploit tools are shown in the book and how to use them. Log Files As stated earlier log files are a way to check events that have been happening on a computer or network. For a hacker, having the ability to change what the log file says can help him not to be noticed. There is code and directions on how to change some log files in the book. Overlooking the Obvious Another sign of a program being hacked is that it will no longer work correctly. Most of the time programs do not work correctly because the hacker has modified them do accomplish another task. A skilled hacker however can modify the program so it still works correctly and does what he wants it do. If a program is exploited there are ways to tell how it happened. Finding out how a program was exploited can be a very tedious process since it usually starts with taking parts of the program and looking at them individually. Putting an exploited program back together again to see how it was exploited is shown in the book. Advanced Camouflage When a hacker is exploiting a program his IP address can be written to a log file. Camouflaging the log files so that his IP address can not be detected is shown in the book. When an IP address is hidden, it is called spoofing the IP address. The Whole Infrastructure The use of intrusion detection systems and intrusion prevention systems greatly helps avoid the risk of being exploited. Even firewalls and routers have log files that can show evidence of hacking. Making sure that outbound TCP connections cannot be processed is one way to limit being found. A few ways are shown in the book on how to use TCP connections so that it is easier to go undetected. Payload Smuggling When using shellcode to exploit programs, it can be caught by intrusion detection systems. Usually the intrusion detection system will catch the programs that are already written and have noticeable shell code in them. Most exploit programs will be caught because real hackers are not using them. There are ways to hide shellcode so it can be harder to detect. A couple of examples on how to hide shellcode are found in the book. Buffer Restrictions Sometimes there are restrictions put on buffers so that vulnerabilities cannot be exploited. There are a few ways that the book depicts on how to get around buffer restrictions. Hardening Countermeasures The exploits that are found in this book have been around for a long time. It took hackers a while before they figured out how to take advantage of the vulnerabilities described in this book. Memory corruption, a change of control, and the use of shellcode are the three easiest steps to exploitation. This an example of a stack and the components of it. Nonexecutable Stack Most applications do not use the stack for any type of executing. One defense is to make the stack non-executable so that buffer overflows cannot be used in the exploitation of the program. This defense is very effective for stopping the use of shellcode in an application. However, there is a way to get around the use of a non-executable stack which is shown and described in the book. Randomized Stack Space A randomized stack is a type of countermeasure used so that the hacker is unable to tell where the shellcode he implemented is. It randomizes the memory layout within the stack. Once again, there is also a way to get around this countermeasure with some examples in the book. 0x700 Cryptology Cryptology is the use of communicating secretly through the use of ciphers, and cryptanalysis is the process of cracking or deciphering such secret communications. This chapter offers information on the theory of cryptology, including the work of Claude Shannon, and concepts including unconditional security, one-time pads, quantum key distribution, and computational security. Using the same key to encrypt and to decrypt messages is symmetric encryption. Asymmetric encryption involves using different keys (public and private). This chapter gives some examples of both kinds of encryption, and how to use them. This an example of how a public and private key is used in the encryption process. A cipher is an encryption algorithm. Combining the use of a symmetric cipher and asymmetric cipher is called a hybrid cipher. Ways to attack ciphers and to get around some encryption methods are shown and described in the book. The chapter also shows methods to figure out encrypted passwords, including brute-force attacks and hash look-ups. It also offers methods to get around wireless 802.11b (Wi-Fi) and wired equivalent privacy (WEP) encryption. See also Computer insecurity Computer security Network security References ^ "Book Review: Hacking". 25 July 2004. Archived from the original on 25 July 2004. Retrieved 26 July 2018. ^ Schaefer, Ed. "Hacking: The Art of Exploitation, 2nd Edition : Linux Magazine". Linux Magazine. Retrieved 26 July 2018. Other sources Erickson, Jon. Hacking: The Art of Exploitation. No Starch Press, 2003. ISBN 1-59327-007-0 John Baichtal (March 3, 2008). "GeekDad Review: Hacking: The Art of Exploitation". Wired. Retrieved March 27, 2009. Retrieved from "