Malware Analysis And Reverse Engineering Cheat Sheet

"Reverse engineering malware has been an integral part of the world of security. At best it has been employed for signature logging malware until now. Since the evolution of new age technologies, this is now being researched as a robust methodology which can lead to more reactive and proactive solutions to the modern security threats that are growing stronger and more sophisticated. This research in its entirety has been an attempt to understand the in and outs of reverse engineering pertaining to malware analysis, with an eye to the future trends in security. Reverse engineering of malware was done with Nugache P2P malware as the target showing that signature based malware identification is ineffective. Developing a proactive approach to quickly identifying malware was the objective that guided this research work. Innovative malware analysis techniques with data mining and rough sets methodologies have been employed in this research work in the quest of a proactive and feasible security solution."--Abstract.

Develop more secure and effective antivirus solutions by leveraging antivirus bypass techniques

Key Features

Gain a clear understanding of the security landscape and research approaches to bypass antivirus software

Become well-versed with practical techniques to bypass antivirus solutions

Discover best practices to develop robust antivirus solutions

Book Description

Antivirus software is built to detect, prevent, and remove malware from systems, but this does not guarantee the security of your antivirus solution as certain changes can trick
the antivirus and pose a risk for users. This book will help you to gain a basic understanding of antivirus software and take you through a series of antivirus bypass techniques that will enable you to bypass antivirus solutions. The book starts by introducing you to the cybersecurity landscape, focusing on cyber threats, malware, and more. You will learn how to collect leads to research antivirus and explore the two common bypass approaches used by the authors. Once you've covered the essentials of antivirus research and bypassing, you'll get hands-on with bypassing antivirus software using obfuscation, encryption, packing, PowerShell, and more. Toward the end, the book covers security improvement recommendations, useful for both antivirus vendors as well as for developers to help strengthen the security and malware detection capabilities of antivirus software. By the end of this security book, you'll have a better understanding of antivirus software and be able to confidently bypass antivirus software. What you will learn Explore the security landscape and get to grips with the fundamentals of antivirus software Discover how to gather AV bypass research leads using malware analysis tools Understand the two commonly used antivirus bypass approaches Find out how to bypass static and dynamic antivirus engines Understand and implement bypass techniques in real-world scenarios Leverage best practices and recommendations for implementing antivirus solutions Who this book is for This book is for security researchers, malware analysts, reverse engineers, pentesters, antivirus vendors looking to strengthen their detection capabilities, antivirus users and companies that want to test and evaluate their antivirus software, organizations that want to test and evaluate antivirus software before purchase or acquisition, and tech-savvy individuals who want to learn new topics. Code reuse detection is a key technique in reverse engineering. However, existing source
code similarity comparison techniques are not applicable to binary code. Moreover, compilers have made this problem even more difficult due to the fact that different assembly code and control flow structures can be generated by the compilers even when implementing the same functionality. To address this problem, we present a fuzzy matching approach to compare two functions. We first obtain our initial mapping between basic blocks by leveraging the concept of longest common subsequence on the basic block level and execution path level. Then, we extend the achieved mapping using neighborhood exploration. To make our approach applicable to large data sets, we designed an effective filtering process using Minhashing and locality-sensitive hashing. Based on the approach proposed in this thesis, we implemented a tool named BinSequence. We conducted extensive experiments to test BinSequence in terms of performance, accuracy, and scalability. Our results suggest that, given a large assembly code repository with millions of functions, BinSequence is efficient and can attain high quality similarity ranking of assembly functions with an accuracy above 90% within seconds. We also present several practical use cases including patch analysis, malware analysis, and bug search. In the use case for patch analysis, we utilized BinSequence to compare the unpatched and patched versions of the same binary, to reveal the vulnerability information and the details of the patch. For this use case, a Windows system driver (HTTP.sys) which contains a recently published critical vulnerability is used. For the malware analysis use case, we utilized BinSequence to identify reused components or already analyzed parts in malware so that the human analyst can focus on those new functionality to save time and effort. In this use case, two infamous malware, Zeus and Citadel, are analyzed. Finally, in the bug search use case, we utilized BinSequence to identify vulnerable functions in software caused by copying and
pasting or sharing buggy source code. In this case, we succeeded in using BinSequence to identify a bug from Firefox. Together, these use cases demonstrate that our tool is both efficient and effective when applied to real-world scenarios. We also compared BinSequence with three state of the art tools: Diaphora, PatchDiff2 and BinDiff. Experiment results show that BinSequence can achieve the best accuracy when compared with these tools.

Malware Analysis is an extremely interesting domain. And like any other specialized domains, it is vast and justly demands considerable time, practice and patience to get started. Malware Analysis Crash Course is a concise & focused book, for those who intend to get started quickly. The book will initiate a student in to the methodology employed in a specimen analysis, processing behavioral and code analysis phases, documenting the observations, tools used in each step of the analysis and importantly setting the mindset steadily with each page. Highly recommended for those who intend to understand the Malware Analysis concepts super quickly, perhaps for the upcoming technical interview for example; and those who wish to learn basics with hands-on, step-by-step example of a specimen analysis.

This book constitutes the refereed proceedings of the 14th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, DIMVA 2017, held in Bonn, Germany, in July 2017. The 18 revised full papers included in this book were carefully reviewed and selected from 67 submissions. They present topics such as enclaves and isolation; malware analysis; cyber-physical systems; detection and protection; code analysis; and web security.

Since 2014 there have been over 120 million new malicious programs registered every year. Due to the amount of new malware appearing every year, analysts have automated large
sections of the malware reverse engineering process. Many automated analysis systems are created by re-implementing analysis techniques rather than automating existing tools that utilize the same techniques. New implementations take longer to create and do not have the same proven quality as a tool that evolved alongside malware for many years. The goal of this study is to assess the efficiency and effectiveness of using existing tools for the application of automated malware analysis. This study focuses on the problem of discovering how malware persists on an infected system. Six tools are chosen based on their usefulness in manual analysis for revealing different persistence techniques employed by malware. The functions of these tools are automated in a fashion that emulates how they can be manually utilized, resulting in information about a tested sample. These six tools are tested against a collection of actual malware samples, pulled from malware families that are known for employing various persistence techniques. The findings are then scanned for indicators of persistence. The results of these tests are used to determine the smallest tool subset that discovers the largest range of persistence mechanisms. For each tool, implementation difficulty is compared to the number of indicators discovered to reveal the effectiveness of similar tools for future analysis applications. The conclusion is that while the tools covered a wide range of persistence mechanisms, the standalone tools that were designed with scripting in mind were more effective than those with multiple system requirements or those with only a graphical interface. It was also discovered that the automation process limits functionality of some tools, as they are designed for analyst interaction. Regaining the tools' functionality lost from automation to use them for other reverse engineering applications could be cumbersome and could require necessary implementation overhauls. Finally, the more successful tools were able to detect a
broader range of techniques, while some less successful tools could only detect a portion of the same techniques. This study concludes that while an analysis system can be created by automating existing tools, the characteristics of the tools chosen impact the workload required to automate them. A well-documented tool that is controllable through a command line interface that offers many configuration options will require less work for an analyst to automate than a tool with little documentation that can only be controlled through a graphical interface.

Rootkits and Bootkits will teach you how to understand and counter sophisticated, advanced threats buried deep in a machine’s boot process or UEFI firmware. With the aid of numerous case studies and professional research from three of the world’s leading security experts, you’ll trace malware development over time from rootkits like TDL3 to present-day UEFI implants and examine how they infect a system, persist through reboot, and evade security software. As you inspect and dissect real malware, you’ll learn:

- How Windows boots—including 32-bit, 64-bit, and UEFI mode—and where to find vulnerabilities
- The details of boot process security mechanisms like Secure Boot, including an overview of Virtual Secure Mode (VSM) and Device Guard
- Reverse engineering and forensic techniques for analyzing real malware, including bootkits like Rovnix/Carberp, Gapz, TDL4, and the infamous rootkits TDL3 and Festi
- How to perform static and dynamic analysis using emulation and tools like Bochs and IDA Pro
- How to better understand the delivery stage of threats against BIOS and UEFI firmware in order to create detection capabilities
- How to use virtualization tools like VMware Workstation to reverse engineer bootkits and the Intel Chipsec tool to dig into forensic analysis

Cybercrime syndicates and malicious actors will continue to write ever more persistent
and covert attacks, but the game is not lost. Explore the cutting edge of malware analysis with Rootkits and Bootkits. Covers boot processes for Windows 32-bit and 64-bit operating systems.

Analyze malicious samples, write reports, and use industry-standard methodologies to confidently triage and analyze adversarial software and malware. Key Features: Investigate, detect, and respond to various types of malware threat. Understand how to use what you've learned as an analyst to produce actionable IOCs and reporting. Explore complete solutions, detailed walkthroughs, and case studies of real-world malware samples.

Book Description: Malicious software poses a threat to every enterprise globally. Its growth is costing businesses millions of dollars due to currency theft as a result of ransomware and lost productivity. With this book, you'll learn how to quickly triage, identify, attribute, and remediate threats using proven analysis techniques.

Malware Analysis Techniques begins with an overview of the nature of malware, the current threat landscape, and its impact on businesses. Once you've covered the basics of malware, you'll move on to discover more about the technical nature of malicious software, including static characteristics and dynamic attack methods within the MITRE ATT&CK framework. You'll also find out how to perform practical malware analysis by applying all that you've learned to attribute the malware to a specific threat and weaponize the adversary's indicators of compromise (IOCs) and methodology against them to prevent them from attacking. Finally, you'll get to grips with common tooling utilized by professional malware analysts and understand the basics of reverse engineering with the NSA's Ghidra platform. By the end of this malware analysis book, you'll be able to perform in-depth static and dynamic analysis and automate key tasks for improved defense against attacks. What you will learn...
Discover how to maintain a safe analysis environment for malware samples. Get to grips with static and dynamic analysis techniques for collecting IOCs. Reverse-engineer and debug malware to understand its purpose. Develop a well-polished workflow for malware analysis. Understand when and where to implement automation to react quickly to threats. Perform malware analysis tasks such as code analysis and API inspection. Who this book is for: This book is for incident response professionals, malware analysts, and researchers who want to sharpen their skillset or are looking for a reference for common static and dynamic analysis techniques. Beginners will also find this book useful to get started with learning about malware analysis. Basic knowledge of command-line interfaces, familiarity with Windows and Unix-like filesystems and registries, and experience in scripting languages such as PowerShell, Python, or Ruby will assist with understanding the concepts covered.

Implement reverse engineering techniques to analyze software, exploit software targets, and defend against security threats like malware and viruses. Key Features: Analyze and improvise software and hardware with real-world examples. Learn advanced debugging and patching techniques with tools such as IDA Pro, x86dbg, and Radare2. Explore modern security techniques to identify, exploit, and avoid cyber threats. Book Description: If you want to analyze software in order to exploit its weaknesses and strengthen its defenses, then you should explore reverse engineering. Reverse Engineering is a hacker-friendly tool used to expose security flaws and questionable privacy practices. In this book, you will learn how to analyse software even without having access to its source code or design documents. You will start off by learning the low-level language used to communicate with the computer and then move on to covering reverse engineering techniques. Next, you will explore analysis techniques using...
real-world tools such as IDA Pro and x86dbg. As you progress through the chapters, you will walk through use cases encountered in reverse engineering, such as encryption and compression, used to obfuscate code, and how to identify and overcome anti-debugging and anti-analysis tricks. Lastly, you will learn how to analyse other types of files that contain code. By the end of this book, you will have the confidence to perform reverse engineering. What you will learn Learn core reverse engineering Identify and extract malware components Explore the tools used for reverse engineering Run programs under non-native operating systems Understand binary obfuscation techniques Identify and analyze anti-debugging and anti-analysis tricks Who this book is for If you are a security engineer or analyst or a system programmer and want to use reverse engineering to improve your software and hardware, this is the book for you. You will also find this book useful if you are a developer who wants to explore and learn reverse engineering. Having some programming/shell scripting knowledge is an added advantage.

A one-of-a-kind guide to setting up a malware research lab, using cutting-edge analysis tools, and reporting the findings Advanced Malware Analysis is a critical resource for every information security professional's anti-malware arsenal. The proven troubleshooting techniques will give an edge to information security professionals whose job involves detecting, decoding, and reporting on malware. After explaining malware architecture and how it operates, the book describes how to create and configure a state-of-the-art malware research lab and gather samples for analysis. Then, you’ll learn how to use dozens of malware analysis tools, organize data, and create metrics-rich reports. A crucial tool for combatting malware—which currently hits each second globally Filled with undocumented methods for
customizing dozens of analysis software tools for very specific uses Leads you through a malware blueprint first, then lab setup, and finally analysis and reporting activities Every tool explained in this book is available in every country around the world

Security Smarts for the Self-Guided IT Professional Learn how to improve the security posture of your organization and defend against some of the most pervasive network attacks. Malware, Rootkits & Botnets: A Beginner's Guide explains the nature, sophistication, and danger of these risks and offers best practices for thwarting them. After reviewing the current threat landscape, the book describes the entire threat lifecycle, explaining how cybercriminals create, deploy, and manage the malware, rootkits, and botnets under their control. You'll learn proven techniques for identifying and mitigating these malicious attacks. Templates, checklists, and examples give you the hands-on help you need to get started protecting your network right away. Malware, Rootkits & Botnets: A Beginner's Guide features: Lingo--Common security terms defined so that you're in the know on the job IMHO--Frank and relevant opinions based on the author's years of industry experience Budget Note--Tips for getting security technologies and processes into your organization's budget In Actual Practice--Exceptions to the rules of security explained in real-world contexts Your Plan--Customizable checklists you can use on the job now Into Action--Tips on how, why, and when to apply new skills and techniques at work

Master malware analysis to protect your systems from getting infected Key Features Set up and model solutions, investigate malware, and prevent it from occurring in future Learn core concepts of dynamic malware analysis, memory forensics, decryption, and much more A practical guide to developing innovative solutions to numerous malware incidents Book
Description With the ever-growing proliferation of technology, the risk of encountering malicious code or malware has also increased. Malware analysis has become one of the most trending topics in businesses in recent years due to multiple prominent ransomware attacks. Mastering Malware Analysis explains the universal patterns behind different malicious software types and how to analyze them using a variety of approaches. You will learn how to examine malware code and determine the damage it can possibly cause to your systems to ensure that it won't propagate any further. Moving forward, you will cover all aspects of malware analysis for the Windows platform in detail. Next, you will get to grips with obfuscation and anti-disassembly, anti-debugging, as well as anti-virtual machine techniques. This book will help you deal with modern cross-platform malware. Throughout the course of this book, you will explore real-world examples of static and dynamic malware analysis, unpacking and decrypting, and rootkit detection. Finally, this book will help you strengthen your defenses and prevent malware breaches for IoT devices and mobile platforms. By the end of this book, you will have learned to effectively analyze, investigate, and build innovative solutions to handle any malware incidents. What you will learn Explore widely used assembly languages to strengthen your reverse-engineering skills Master different executable file formats, programming languages, and relevant APIs used by attackers Perform static and dynamic analysis for multiple platforms and file types Get to grips with handling sophisticated malware cases Understand real advanced attacks, covering all stages from infiltration to hacking the system Learn to bypass anti-reverse engineering techniques Who this book is for If you are an IT security administrator, forensic analyst, or malware researcher looking to secure against malicious software or investigate malicious code, this book is for you. Prior programming
experience and a fair understanding of malware attacks and investigation is expected. Android Security: Attacks and Defenses is for anyone interested in learning about the strengths and weaknesses of the Android platform from a security perspective. Starting with an introduction to Android OS architecture and application programming, it will help readers get up to speed on the basics of the Android platform and its security issues. Explaining the Android security model and architecture, the book describes Android permissions, including Manifest permissions, to help readers analyze applications and understand permission requirements. It also rates the Android permissions based on security implications and covers JEB Decompiler. The authors describe how to write Android bots in JAVA and how to use reversing tools to decompile any Android application. They also cover the Android file system, including import directories and files, so readers can perform basic forensic analysis on file system and SD cards. The book includes access to a wealth of resources on its website: www.androidinsecurity.com. It explains how to crack SecureApp.apk discussed in the text and also makes the application available on its site. The book includes coverage of advanced topics such as reverse engineering and forensics, mobile device pen-testing methodology, malware analysis, secure coding, and hardening guidelines for Android. It also explains how to analyze security implications for Android mobile devices/applications and incorporate them into enterprise SDLC processes. The book’s site includes a resource section where readers can access downloads for applications, tools created by users, and sample applications created by the authors under the Resource section. Readers can easily download the files and use them in conjunction with the text, wherever needed. Visit www.androidinsecurity.com for more information.
Has the GIAC Reverse Engineering Malware work been fairly and/or equitably divided and delegated among team members who are qualified and capable to perform the work? Has everyone contributed? How do we Identify specific GIAC Reverse Engineering Malware investment and emerging trends? What about GIAC Reverse Engineering Malware Analysis of results? Will team members regularly document their GIAC Reverse Engineering Malware work? In the case of a GIAC Reverse Engineering Malware project, the criteria for the audit derive from implementation objectives. an audit of a GIAC Reverse Engineering Malware project involves assessing whether the recommendations outlined for implementation have been met. in other words, can we track that any GIAC Reverse Engineering Malware project is implemented as planned, and is it working? Defining, designing, creating, and implementing a process to solve a business challenge or meet a business objective is the most valuable role... In EVERY company, organization and department. Unless you are talking a one-time, single-use project within a business, there should be a process. Whether that process is managed and implemented by humans, AI, or a combination of the two, it needs to be designed by someone with a complex enough perspective to ask the right questions. Someone capable of asking the right questions and step back and say, 'What are we really trying to accomplish here? And is there a different way to look at it?' For more than twenty years, The Art of Service's Self-Assessments empower people who can do just that - whether their title is marketer, entrepreneur, manager, salesperson, consultant, business process manager, executive assistant, IT Manager, CxO etc... - they are the people who rule the future. They are people who watch the process as it happens, and ask the right questions to make the process work better. This book is for managers, advisors, consultants, specialists, professionals and
anyone interested in GIAC Reverse Engineering Malware assessment. All the tools you need
to an in-depth GIAC Reverse Engineering Malware Self-Assessment. Featuring 488 new and
updated case-based questions, organized into seven core areas of process design, this Self-
Assessment will help you identify areas in which GIAC Reverse Engineering Malware
improvements can be made. In using the questions you will be better able to: - diagnose GIAC
Reverse Engineering Malware projects, initiatives, organizations, businesses and processes
using accepted diagnostic standards and practices - implement evidence-based best practice
strategies aligned with overall goals - integrate recent advances in GIAC Reverse Engineering
Malware and process design strategies into practice according to best practice guidelines
Using a Self-Assessment tool known as the GIAC Reverse Engineering Malware Scorecard,
you will develop a clear picture of which GIAC Reverse Engineering Malware areas need
attention. Included with your purchase of the book is the GIAC Reverse Engineering Malware
Self-Assessment downloadable resource, which contains all questions and Self-Assessment
areas of this book in a ready to use Excel dashboard, including the self-assessment, graphic
insights, and project planning automation - all with examples to get you started with the
assessment right away. Access instructions can be found in the book. You are free to use the
Self-Assessment contents in your presentations and materials for customers without asking us
- we are here to help.

Nearly every business depends on its network to provide information services to carry out
essential activities, and network intrusion attacks have been growing increasingly frequent and
severe. When network intrusions do occur, it’s imperative that a thorough and systematic
analysis and investigation of the attack is conducted to determine the nature of the threat and
the extent of information lost, stolen, or damaged during the attack. A thorough and timely investigation and response can serve to minimize network downtime and ensure that critical business systems are maintained in full operation. *Network Intrusion Analysis* teaches the reader about the various tools and techniques to use during a network intrusion investigation. The book focuses on the methodology of an attack as well as the investigative methodology, challenges, and concerns. This is the first book that provides such a thorough analysis of network intrusion investigation and response. *Network Intrusion Analysis* addresses the entire process of investigating a network intrusion by: *Providing a step-by-step guide to the tools and techniques used in the analysis and investigation of a network intrusion. *Providing real-world examples of network intrusions, along with associated workarounds. *Walking you through the methodology and practical steps needed to conduct a thorough intrusion investigation and incident response, including a wealth of practical, hands-on tools for incident assessment and mitigation.*

*Network Intrusion Analysis* addresses the entire process of investigating a network intrusion:

- Provides a step-by-step guide to the tools and techniques used in the analysis and investigation of a network intrusion.
- Provides real-world examples of network intrusions, along with associated workarounds.
- Walks readers through the methodology and practical steps needed to conduct a thorough intrusion investigation and incident response, including a wealth of practical, hands-on tools for incident assessment and mitigation.

Master the fundamentals of malware analysis for the Windows platform and enhance your anti-malware skill set.

About This Book

Set the baseline towards performing malware analysis on the Windows platform and how to use the tools required to deal with malware. Understand how to decipher x86 assembly code from source code inside your favourite development
environment A step-by-step based guide that reveals malware analysis from an industry insider and demystifies the process Who This Book Is For This book is best for someone who has prior experience with reverse engineering Windows executables and wants to specialize in malware analysis. The book presents the malware analysis thought process using a show-and-tell approach, and the examples included will give any analyst confidence in how to approach this task on their own the next time around. What You Will Learn Use the positional number system for clear conception of Boolean algebra, that applies to malware research purposes Get introduced to static and dynamic analysis methodologies and build your own malware lab Analyse destructive malware samples from the real world (ITW) from fingerprinting and static/dynamic analysis to the final debrief Understand different modes of linking and how to compile your own libraries from assembly code and integrate the code into your final program Get to know about the various emulators, debuggers and their features, and sandboxes and set them up effectively depending on the required scenario Deal with other malware vectors such as pdf and MS-Office based malware as well as scripts and shellcode In Detail Windows OS is the most used operating system in the world and hence is targeted by malware writers. There are strong ramifications if things go awry. Things will go wrong if they can, and hence we see a salvo of attacks that have continued to disrupt the normal scheme of things in our day to day lives. This book will guide you on how to use essential tools such as debuggers, disassemblers, and sandboxes to dissect malware samples. It will expose your innards and then build a report of their indicators of compromise along with detection rule sets that will enable you to help contain the outbreak when faced with such a situation. We will start with the basics of computing fundamentals such as number systems and Boolean algebra. Further,
you'll learn about x86 assembly programming and its integration with high level languages such as C++. You'll understand how to decipher disassembly code obtained from the compiled source code and map it back to its original design goals. By delving into end to end analysis with real-world malware samples to solidify your understanding, you'll sharpen your technique of handling destructive malware binaries and vector mechanisms. You will also be encouraged to consider analysis lab safety measures so that there is no infection in the process. Finally, we'll have a rounded tour of various emulations, sandboxing, and debugging options so that you know what is at your disposal when you need a specific kind of weapon in order to nullify the malware.

Style and approach
An easy to follow, hands-on guide with descriptions and screenshots that will help you execute effective malicious software investigations and conjure up solutions creatively and confidently.

Become well-versed with forensics for the Android, iOS, and Windows 10 mobile platforms by learning essential techniques and exploring real-life scenarios

Key Features
Apply advanced forensic techniques to recover deleted data from mobile devices
Retrieve and analyze data stored not only on mobile devices but also on the cloud and other connected mediums
Use the power of mobile forensics on popular mobile platforms by exploring different tips, tricks, and techniques

Book Description
Mobile phone forensics is the science of retrieving data from a mobile phone under forensically sound conditions. This updated fourth edition of Practical Mobile Forensics delves into the concepts of mobile forensics and its importance in today's world. The book focuses on teaching you the latest forensic techniques to investigate mobile...
devices across various mobile platforms. You will learn forensic techniques for multiple OS versions, including iOS 11 to iOS 13, Android 8 to Android 10, and Windows 10. The book then takes you through the latest open source and commercial mobile forensic tools, enabling you to analyze and retrieve data effectively. From inspecting the device and retrieving data from the cloud, through to successfully documenting reports of your investigations, you'll explore new techniques while building on your practical knowledge. Toward the end, you will understand the reverse engineering of applications and ways to identify malware. Finally, the book guides you through parsing popular third-party applications, including Facebook and WhatsApp. By the end of this book, you will be proficient in various mobile forensic techniques to analyze and extract data from mobile devices with the help of open source solutions. What you will learn

Discover new data extraction, data recovery, and reverse engineering techniques in mobile forensics
Understand iOS, Windows, and Android security mechanisms
Identify sensitive files on every mobile platform
Extract data from iOS, Android, and Windows platforms
Understand malware analysis, reverse engineering, and data analysis of mobile devices
Explore various data recovery techniques on all three mobile platforms

Who this book is for
This book is for forensic examiners with basic experience in mobile forensics or open source solutions for mobile forensics. Computer security professionals, researchers or anyone looking to gain a deeper understanding of mobile internals will also find this book useful. Some underst...
Analyzing how hacks are done, so as to stop them in the future. Reverse engineering is the process of analyzing hardware or software and understanding it, without having access to the source code or design documents. Hackers are able to reverse engineer systems and exploit what they find with scary results. Now the good guys can use the same tools to thwart these threats. Practical Reverse Engineering goes under the hood of reverse engineering for security analysts, security engineers, and system programmers, so they can learn how to use these same processes to stop hackers in their tracks. The book covers x86, x64, and ARM (the first book to cover all three); Windows kernel-mode code rootkits and drivers; virtual machine protection techniques; and much more. Best of all, it offers a systematic approach to the material, with plenty of hands-on exercises and real-world examples. Offers a systematic approach to understanding reverse engineering, with hands-on exercises and real-world examples. Covers x86, x64, and advanced RISC machine (ARM) architectures as well as deobfuscation and virtual machine protection techniques. Provides special coverage of Windows kernel-mode code (rootkits/drivers), a topic not often covered elsewhere, and explains how to analyze drivers step by step. Demystifies topics that have a steep learning curve. Includes a bonus chapter on reverse engineering tools. Practical Reverse Engineering: Using x86, x64, ARM, Windows Kernel, and Reversing Tools provides crucial, up-to-date guidance for a broad range of IT professionals. This training course is a Linux version of the previous Practical Foundations of
Windows Debugging, Disassembly, Reversing book. It also complements Accelerated Linux Core Dump Analysis training course. Although the book skeleton is the same as its Windows predecessor, the content was revised entirely because of a different operating system, debugger (GDB), toolchain (GCC, assembler, linker), application binary interface, and even an assembly language flavor, AT&T. The course is useful for: Software technical support and escalation engineers Software engineers coming from JVM background Software testers Engineers coming from non-Linux environments, for example, Windows or Mac OS X Linux C/C++ software engineers without assembly language background Security researchers without assembly language background Beginners learning Linux software reverse engineering techniques This book can also be used as x64 assembly language and Linux debugging supplement for relevant undergraduate level courses.

Malware analysis is big business, and attacks can cost a company dearly. When malware breaches your defenses, you need to act quickly to cure current infections and prevent future ones from occurring. For those who want to stay ahead of the latest malware, Practical Malware Analysis will teach you the tools and techniques used by professional analysts. With this book as your guide, you'll be able to safely analyze, debug, and disassemble any malicious software that comes your way. You'll learn how to: –Set up a safe virtual environment to analyze malware –Quickly extract network signatures and host-based indicators –Use key analysis tools like IDA Pro, OllyDbg,
and WinDbg—Overcome malware tricks like obfuscation, anti-disassembly, anti-debugging, and anti-virtual machine techniques—Use your newfound knowledge of Windows internals for malware analysis—Develop a methodology for unpacking malware and get practical experience with five of the most popular packers—Analyze special cases of malware with shellcode, C++, and 64-bit code Hands-on labs throughout the book challenge you to practice and synthesize your skills as you dissect real malware samples, and pages of detailed dissections offer an over-the-shoulder look at how the pros do it. You'll learn how to crack open malware to see how it really works, determine what damage it has done, thoroughly clean your network, and ensure that the malware never comes back. Malware analysis is a cat-and-mouse game with rules that are constantly changing, so make sure you have the fundamentals. Whether you're tasked with securing one network or a thousand networks, or you're making a living as a malware analyst, you'll find what you need to succeed in Practical Malware Analysis.

Understand malware analysis and its practical implementation Key Features Explore the key concepts of malware analysis and memory forensics using real-world examples Learn the art of detecting, analyzing, and investigating malware threats Understand adversary tactics and techniques Book Description Malware analysis and memory forensics are powerful analysis and investigation techniques used in reverse engineering, digital forensics, and incident response. With adversaries becoming
sophisticated and carrying out advanced malware attacks on critical infrastructures, data centers, and private and public organizations, detecting, responding to, and investigating such intrusions is critical to information security professionals. Malware analysis and memory forensics have become must-have skills to fight advanced malware, targeted attacks, and security breaches. This book teaches you the concepts, techniques, and tools to understand the behavior and characteristics of malware through malware analysis. It also teaches you techniques to investigate and hunt malware using memory forensics. This book introduces you to the basics of malware analysis, and then gradually progresses into the more advanced concepts of code analysis and memory forensics. It uses real-world malware samples, infected memory images, and visual diagrams to help you gain a better understanding of the subject and to equip you with the skills required to analyze, investigate, and respond to malware-related incidents. What you will learn

- Create a safe and isolated lab environment for malware analysis
- Extract the metadata associated with malware
- Determine malware's interaction with the system
- Perform code analysis using IDA Pro and x64dbg
- Reverse-engineer various malware functionalities
- Reverse engineer and decode common encoding/encryption algorithms
- Reverse-engineer malware code injection and hooking techniques
- Investigate and hunt malware using memory forensics

Who this book is for

This book is for incident responders, cyber-security investigators, system administrators, malware analyst, forensic practitioners, student, or curious security professionals.
professionals interested in learning malware analysis and memory forensics. Knowledge of programming languages such as C and Python is helpful but is not mandatory. If you have written few lines of code and have a basic understanding of programming concepts, you'll be able to get most out of this book. Can we do GIAC Reverse Engineering Malware without complex (expensive) analysis? How do you use GIAC Reverse Engineering Malware data and information to support organizational decision making and innovation? Are there any specific expectations or concerns about the GIAC Reverse Engineering Malware team, GIAC Reverse Engineering Malware itself? How did the GIAC Reverse Engineering Malware manager receive input to the development of a GIAC Reverse Engineering Malware improvement plan and the estimated completion dates/times of each activity? Who will be responsible for documenting the GIAC Reverse Engineering Malware requirements in detail? Defining, designing, creating, and implementing a process to solve a business challenge or meet a business objective is the most valuable role... In EVERY company, organization and department. Unless you are talking a one-time, single-use project within a business, there should be a process. Whether that process is managed and implemented by humans, AI, or a combination of the two, it needs to be designed by someone with a complex enough perspective to ask the right questions. Someone capable of asking the right questions and step back and say, 'What are we really trying to accomplish here? And is there a different way to look at it?' This Self-Assessment
empowers people to do just that - whether their title is entrepreneur, manager, consultant, (Vice-)President, CxO etc... - they are the people who rule the future. They are the person who asks the right questions to make GIAC Reverse Engineering Malware investments work better. This GIAC Reverse Engineering Malware All-Inclusive Self-Assessment enables You to be that person. All the tools you need to an in-depth GIAC Reverse Engineering Malware Self-Assessment. Featuring 488 new and updated case-based questions, organized into seven core areas of process design, this Self-Assessment will help you identify areas in which GIAC Reverse Engineering Malware improvements can be made. In using the questions you will be better able to: - diagnose GIAC Reverse Engineering Malware projects, initiatives, organizations, businesses and processes using accepted diagnostic standards and practices - implement evidence-based best practice strategies aligned with overall goals - integrate recent advances in GIAC Reverse Engineering Malware and process design strategies into practice according to best practice guidelines Using a Self-Assessment tool known as the GIAC Reverse Engineering Malware Scorecard, you will develop a clear picture of which GIAC Reverse Engineering Malware areas need attention. Your purchase includes access details to the GIAC Reverse Engineering Malware self-assessment dashboard download which gives you your dynamically prioritized projects-ready tool and shows your organization exactly what to do next. Your exclusive instant access details can be found in your book.
An opaque predicate is a predicate whose value is known to the obfuscator but is difficult to deduce. It can be seamlessly applied together with other obfuscation methods such as junk code to turn reverse engineering attempts into arduous work. Opaque predicates have been widely used in various areas of software security such as software protection, software watermarking, obfuscation, and metamorphic malware. The arms race between the construction and detection of opaque predicates is an interesting topic in computer security area. This thesis introduces new attack and defense techniques about opaque predicates in binary code. First, a logic oriented opaque predicate detection tool called LOOP is proposed. By conducting symbolic execution along a trace, LOOP constructs general logical formulas to represent the intrinsic characteristics of opaque predicates. The formulas are then solved by a constraint solver and the result answers whether the predicate under examination is opaque or not. Besides, LOOP is obfuscation resilient and able to detect previously unknown opaque predicates. Our experimental result demonstrates LOOP is effective and efficient. By integrating LOOP with code normalization for matching metamorphic malware variants, we show that LOOP is an appealing complement to existing malware defenses. Second, a new control flow obfuscation scheme called generalized dynamic opaque predicate is proposed. We extend the conventional concept of dynamic opaque predicate to common program structures (e.g., straight-line code, branch, and loop). Besides, our new design does not require dynamic opaque predicates to be strictly
adjacent, which is more resilient to deobfuscation techniques. The evaluation result shows generalized dynamic opaque predicates overcome the limitations in conventional opaque predicates. Third, we propose a novel technique called bit-precise symbolic loop mapping to identify cryptographic functions in obfuscated binary code. Advanced opaque predicates adopt cryptographic functions to challenge existing opaque predicate detection methods. Our trace-based approach captures the semantics of possible cryptographic algorithms with bit-precise symbolic execution in a loop. Then we perform guided fuzzing to efficiently match boolean formulas with known reference implementations. We have developed a prototype called CryptoHunt and evaluated it with a set of obfuscated synthetic examples, well-known cryptographic libraries, and malware. Compared with the existing tools, CryptoHunt is a general approach to detecting commonly used cryptographic functions such as TEA, AES, RC4, MD5, and RSA under different control and data obfuscation scheme combinations. Our research deepens and expands the knowledge of attack and defense techniques about opaque predicates. We are the first to give a systematic categorization of opaque predicates and a method for the detection of dynamic opaque predicates. Our research about the extension of dynamic opaque predicates greatly widens its application. The cryptographic algorithm detection can be applied to various fields, such as ransomware detection, malware analysis, and software reverse engineering. The rapid growth and development of Android-based devices has resulted in a wealth
of sensitive information on mobile devices that offer minimal malware protection. This has created an immediate need for security professionals that understand how to best approach the subject of Android malware threats and analysis. In Android Malware and Analysis, Ken Dunham, renowned global malware expert and author, teams up with international experts to document the best tools and tactics available for analyzing Android malware. The book covers both methods of malware analysis: dynamic and static. This tactical and practical book shows you how to use dynamic malware analysis to check the behavior of an application/malware as it has been executed in the system. It also describes how you can apply static analysis to break apart the application/malware using reverse engineering tools and techniques to recreate the actual code and algorithms used. The book presents the insights of experts in the field, who have already sized up the best tools, tactics, and procedures for recognizing and analyzing Android malware threats quickly and effectively. You also get access to an online library of tools that supplies what you will need to begin your own analysis of Android malware threats. Tools available on the book’s site include updated information, tutorials, code, scripts, and author assistance. This is not a book on Android OS, fuzz testing, or social engineering. Instead, it is about the best ways to analyze and tear apart Android malware threats. After reading the book, you will be able to immediately implement the tools and tactics covered to identify and analyze the latest evolution of Android threats. Updated information, tutorials, a private forum, code,
scripts, tools, and author assistance are available at AndroidRisk.com for first-time owners of the book.

Malware Analysis and Detection Engineering A Comprehensive Approach to Detect and Analyze Modern Malware Apress

Detect potential bugs in your code or program and develop your own tools using the Ghidra reverse engineering framework developed by the NSA project Key Features

- Make the most of Ghidra on different platforms such as Linux, Windows, and macOS
- Leverage a variety of plugins and extensions to perform disassembly, assembly, decompilation, and scripting
- Discover how you can meet your cybersecurity needs by creating custom patches and tools

Book Description

Ghidra, an open source software reverse engineering (SRE) framework created by the NSA research directorate, enables users to analyze compiled code on any platform, whether Linux, Windows, or macOS. This book is a starting point for developers interested in leveraging Ghidra to create patches and extend tool capabilities to meet their cybersecurity needs. You'll begin by installing Ghidra and exploring its features, and gradually learn how to automate reverse engineering tasks using Ghidra plug-ins. You'll then see how to set up an environment to perform malware analysis using Ghidra and how to use it in the headless mode. As you progress, you'll use Ghidra scripting to automate the task of identifying vulnerabilities in executable binaries. The book also covers advanced topics such as developing Ghidra plug-ins, developing your own GUI, incorporating new process architectures if needed, and contributing to the Ghidra project. By the end of this Ghidra book, you'll have developed the skills you need to harness the power of Ghidra for analyzing and avoiding potential vulnerabilities in code and networks. What you will learn

Get to grips with using
Ghidra's features, plug-ins, and extensions Understand how you can contribute to Ghidra
Focus on reverse engineering malware and perform binary auditing Automate reverse
engineering tasks with Ghidra plug-ins Become well-versed with developing your own Ghidra
extensions, scripts, and features Automate the task of looking for vulnerabilities in executable
binaries using Ghidra scripting Find out how to use Ghidra in the headless mode Who this
book is for This SRE book is for developers, software engineers, or any IT professional with
some understanding of cybersecurity essentials. Prior knowledge of Java or Python, along with
experience in programming or developing applications, is required before getting started with
this book.
Software reverse engineering is a complex process that incorporates different techniques
involving static and dynamic analyses of software programs. Numerous tools are available that
help reverse engineers in automating the dynamic analysis process. However, the process of
static analysis remains a challenging and tedious process for reverse engineers. The static
analysis process requires a great amount of manual work. Therefore, it is very demanding and
time-consuming. One aspect of reverse engineering that provides reverse engineers with
useful information regarding a statically analyzed piece of code is function fingerprinting.
Binary code fingerprinting is a challenging problem that requires an in-depth analysis of
internal binary code components for deriving identifiable and expressive signatures. Binary
code fingerprints are helpful in the reverse engineering process and have various security
applications such as malware variant detection, malware clustering, binary auditing, function
recognition, and library identification. Moreover, binary code fingerprinting is also useful in
automating some reverse engineering tasks such as clone detection, library function
identification, code similarity, authorship attribution, etc. In addition, code fingerprints are valuable in cyber forensics as well as the process of patch analysis in order to identify patches or make sure that the patch complies with the security requirements. In this thesis, we propose a binary function fingerprinting and matching approach and implement a tool named BinSign based on the proposed approach that enhances and accelerates the reverse engineering process. The main objective of BinSign is to provide an accurate and scalable solution to binary code fingerprinting by computing and matching structural and syntactic code profiles for disassemblies while outperforming existing techniques. The structural profile of binary code is captured through decomposing the control-flow-graph of a function into tracelets. We describe the underlying methodology and evaluate its performance in several use cases, including function matching, function reuse, library function detection, malware analysis, and function indexing scalability. We also provide some insights into the effects of different optimization levels and obfuscation techniques on our fingerprint matching methodology. Additionally, we emphasize the scalability aspect of BinSign that is achieved through applying locality sensitive hashing, filtering techniques, and distributing the computations across several machines. The min-hashing process is combined with the banding technique of locality sensitive hashing in order to ensure a scalable and efficient fingerprint matching process. We perform our experiments on a database of 6 million functions that includes well-known libraries, malware samples, and some dynamic library files obtained from the Microsoft Windows operating system. The indexing process of fingerprints is distributed across multiple machines and it requires an average time of 0.0072 seconds per function. A comparison is also conducted with relevant existing tools, which shows that BinSign achieves a higher accuracy than these tools.
This book constitutes the refereed proceedings of the 15th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, DIMVA 2018, held in Saclay, France, in June 2018. The 17 revised full papers and 1 short paper included in this book were carefully reviewed and selected from 59 submissions. They present topics such as malware analysis; mobile and embedded security; attacks; detection and containment; web and browser security; and reverse engineering.

When it comes to cybersecurity, everyone needs to be part of the solution if we ever hope to slow the rising tide of cyberattacks. Nearly 4.5 billion people—about 60% of the world’s population—were actively online last year. Every one of these individuals conducted business, shopped, handled their finances or browsed for information using a computer, tablet, smartphone or some other connected device at home or work. But while greater global connectivity brings a wealth of benefits, we often fail to recognize that all of these connected people pose a potential cyberthreat to themselves and those around them. As consumers, we have reached an important crossroads; we want high-tech companies and government agencies to protect us from cyberthreats, yet we, too, bear responsibility for securing our connected systems and data. If we ever hope to slow the rising tide of cyberattacks, everyone needs to be part of the solution.

The process of software reverse engineering and malware analysis often comprise a combination of static and dynamic analyses. The successful outcome of each step is tightly coupled with the functionalities of the tools and skills of the reverse engineer. Even though automated tools are available for dynamic analysis, the static analysis process is a fastidious and time-consuming task as it requires manual work and strong expertise in assembly coding.
In order to enhance and accelerate the reverse engineering process, we introduce a new dimension known as clone-based analysis. Recently, binary clone matching has been studied with a focus on detecting assembly (binary) clones. An alternative approach in clone analysis, which is studied in the present research, is concerned with assembly to source code matching. There are two major advantages in considering this extra dimension. The first advantage is to avoid dealing with low-level assembly code in situations where the corresponding high-level code is available. The other advantage is to prevent reverse engineering parts of the software that have been analyzed before. The clone-based analysis can be helpful in significantly reducing the required time and improving the accuracy of static analysis. In this research, we elaborate a framework for assembly to open-source code matching. Two types of analyses are provided by the framework, namely online and offline. The online analysis process triggers queries to online source code repositories based on extracted features from the functions at the assembly level. The result is the matched set of references to the open-source project files with similar features. Moreover, the offline analysis assigns functionality tags and provides in-depth information regarding the potential functionality of a portion of the assembly file. It reports on function stack frames, prototypes, arguments, variables, return values and low-level system calls. Besides, the offline analysis is based on a built-in dictionary of common user-level and kernel-level API functions that are used by malware to interact with the operating system. These functions are called for performing tasks such as file I/O, network communications, registry modification, and service manipulation. The offline analysis process has been expanded through an incremental learning mechanism which results in an improved detection of crypto-related functions in the disassembly. The other developed extension is a
customized local code repository which performs automated source code parsing, feature extraction, and dataset generation for code matching. We apply the framework in several reverse engineering and malware analysis scenarios. Also, we show that the underlying tools and techniques are effective in providing additional insights into the functionality, inner workings, and components of the target binaries.

Improvement in technology has inevitably altered the tactic of criminals to thievery. In recent times, information is the real commodity and it is thus subject to theft as any other possessions: cryptocurrency, credit card numbers, and illegal digital material are on the top. If globally available platforms for smartphones are considered, the Android open source platform (AOSP) emerges as a prevailing contributor to the market and its popularity continues to intensify. Whilst it is beneficiary for users, this development simultaneously makes a prolific environment for exploitation by immoral developers who create malware or reuse software illegitimately acquired by reverse engineering. Android malware analysis techniques are broadly categorized into static and dynamic analysis. Many researchers have also used feature-based learning to build and sustain working security solutions. Although Android has its base set of permissions in place to protect the device and resources, it does not provide strong enough security framework to defend against attacks. This thesis presents several contributions in the domain of security of Android applications and the data within these applications. First, a brief survey of threats, vulnerability and security analysis tools for the AOSP is presented. Second, we develop and use a genre extraction algorithm for Android applications to check the availability of those applications in Google Play Store. Third, an algorithm for extracting unclaimed permissions is proposed which will give a set of
unnecessary permissions for applications under examination. Finally, machine learning aided approaches for analysis of Android malware were adopted. Features including permissions, APIs, content providers, broadcast receivers, and services are extracted from benign (~2,000) and malware (5,560) applications and examined for evaluation. We create feature vector combinations using these features and feed these vectors to various classifiers. Based on the evaluation metrics of classifiers, we scrutinize classifier performance with respect to specific feature combination. Classifiers such as SVM, Logistic Regression and Random Forests spectacle a good performance whilst the dataset of combination of permissions and APIs records the maximum accuracy for Logistic Regression.

Beginning with a basic primer on reverse engineering-including computer internals, operating systems, and assembly language-and then discussing the various applications of reverse engineering, this book provides readers with practical, in-depth techniques for software reverse engineering. The book is broken into two parts, the first deals with security-related reverse engineering and the second explores the more practical aspects of reverse engineering. In addition, the author explains how to reverse engineer a third-party software library to improve interfacing and how to reverse engineer a competitor's software to build a better product. * The first popular book to show how software reverse engineering can help defend against security threats, speed up development, and unlock the secrets of competitive products * Helps developers plug security holes by demonstrating how hackers exploit reverse engineering techniques to crack copy-protection schemes and identify software targets for viruses and other malware * Offers a primer on advanced reverse-engineering, delving into "disassembly"-code-level reverse engineering-and explaining how to decipher assembly
The reverse engineering of binaries is a tedious and time consuming task, yet mandatory when the need arises to understand the behaviour of a program for which source code is unavailable. Instances of source code loss for old arcade games [1] and the steadily growing amount of malware [2] are prominent use cases requiring reverse engineering. One of the challenges when dealing with binaries is the loss of low level type information, i.e., primitive and compound types, which even state-of-the-art type recovery tools often cannot reconstruct with full accuracy. Further programmers most commonly use high level data structures, such as linked lists, in addition to primitive types. Therefore detection of dynamic data structure shapes is an important aspect of reverse engineering. Though the recognition of dynamic data structure shapes in the presence of tricky programming concepts such as pointer arithmetic and casts - which are both fundamental concepts to enable, e.g., the frequently used Linux kernel list [3] - also bring current shape detection tools to their limits. A recent approach called Data Structure Investigator (DSI) [4], aims for the detection of dynamic pointer based data structures. While the approach is general in nature, a concrete realization for C programs requiring source code is envisioned as programming constructs such as type casts and pointer arithmetic will stress test the approach. Therefore, the first research question addressed in this dissertation is whether DSI can meet its goal in the presence of the sheer multitude of existing data structure implementations. The second research
question is whether DSI can be opened up to reverse engineer C/C++ binaries, even in the presence of type information loss and the variety of C/C++ programming constructs. Both questions are answered positively in this dissertation. The first is answered by realizing the DSI source code approach, which requires detailing fundamental aspects of DSI's theory to arrive at a working implementation, e.g., handling the consistency of DSI's memory abstraction and quantifying the interconnections found within a dynamic pointer based data structure, e.g., a parent child nesting scenario, to allow for its detection. DSI's utility is evaluated on an extensive benchmark including real world examples (libusb [5], bash [6]) and shape analysis examples, [7,8]. The second question is answered through the development of a DSI prototype for binaries (DSIbin). To compensate for the loss of perfect type information found in source code, DSIbin interfaces with the state-of-the-art type recovery tool Howard [9]. Notably, DSIbin improves upon type information recovered by Howard. This is accomplished through a much improved nested struct detection and type merging algorithm, both of which are fundamental aspects for the reverse engineering of binaries. The proposed approach is again evaluated by a diverse benchmark containing real world examples such as, the VNC clipping library, The Computer Language Benchmarks Game and the Olden Benchmark, as well as examples taken from the shape analysis literature. In summary, this dissertation improves upon the state-of-the-art of shape detection and reverse engineering by (i) realizing and evaluating the DSI approach, which includes
contributing to DSI's theory and results in the DSI prototype; (ii) opening up DSI for C/C++ binaries so as to extend DSI to reverse engineering, resulting in the DSIbin prototype; (iii) handling data structures with DSIbin not covered by some related work such as skip lists; (iv) refining the nesting detection and performing type merging for types excavated by Howard. Further, DSIbin's ultimate future use case of malware analysis is hardened by revealing the presence of dynamic data structures in multiple real world malware samples. In summary, this dissertation advanced the dynamic analysis of data structure shapes with the aforementioned contributions to the DSI approach for source code and further by transferring this new technology to the analysis of binaries. The latter resulted in the additional insight that high level dynamic data structure information can help to infer low level type information.

"Software security researchers commonly reverse engineer and analyze current malicious software (malware) to determine what the latest techniques malicious attackers are utilizing and how to protect computer systems from attack. The most common analysis methods involve examining how the program behaves during execution and interpreting its machine-level instructions. However, modern malicious applications use advanced anti-debugger, anti-virtualization, and code packing techniques to obfuscate the malware's true activities and divert security analysts. Malware analysts currently do not have a simple method for tracing malicious code activity at the instruction-level in a highly undetectable environment. There also lacks a
simple method for combining actual run-time register and memory values with statically disassembled code. Combining statically disassembled code with the run-time values found in the memory and registers being accessed would create a new level of analysis possible by combining key aspects of static analysis with dynamic analysis. This thesis presents EtherAnnotate, a new extension to the Xen Ether virtualization framework and the IDA Pro disassembler to aid in the task of malicious software analysis. This new extension consists of two separate components - an enhanced instruction tracer and a graphical annotation and visualization plug-in for IDA Pro. The specialized instruction tracer places a malware binary into a virtualized environment and records the contents of all processor general register values that occur during its execution. The annotation plug-in for IDA Pro interprets the output of the instruction tracer and adds line comments of the register values in addition to visualizing code coverage of all disassembled instructions that were executed during the malware's execution. These two tools can be combined to provide a new level of introspection for advanced malware that was not available with the previous state-of-the-art analysis tools"--Abstract, leaf iii.

Malware Forensics: Investigating and Analyzing Malicious Code covers the complete process of responding to a malicious code incident. Written by authors who have investigated and prosecuted federal malware cases, this book deals with the emerging and evolving field of live forensics, where investigators examine a computer system to
collect and preserve critical live data that may be lost if the system is shut down. Unlike other forensic texts that discuss live forensics on a particular operating system, or in a generic context, this book emphasizes a live forensics and evidence collection methodology on both Windows and Linux operating systems in the context of identifying and capturing malicious code and evidence of its effect on the compromised system. It is the first book detailing how to perform live forensic techniques on malicious code. The book gives deep coverage on the tools and techniques of conducting runtime behavioral malware analysis (such as file, registry, network and port monitoring) and static code analysis (such as file identification and profiling, strings discovery, armoring/packing detection, disassembling, debugging), and more. It explores over 150 different tools for malware incident response and analysis, including forensic tools for preserving and analyzing computer memory. Readers from all educational and technical backgrounds will benefit from the clear and concise explanations of the applicable legal case law and statutes covered in every chapter. In addition to the technical topics discussed, this book also offers critical legal considerations addressing the legal ramifications and requirements governing the subject matter. This book is intended for system administrators, information security professionals, network personnel, forensic examiners, attorneys, and law enforcement working with the inner-workings of computer memory and malicious code. * Winner of Best Book Bejtlich read in 2008! * http://taosecurity.blogspot.com/2008/12/best-book-bejtlich-read-in-2008.html
Authors have investigated and prosecuted federal malware cases, which allows them to provide unparalleled insight to the reader. First book to detail how to perform "live forensic" techniques on malicious code. In addition to the technical topics discussed, this book also offers critical legal considerations addressing the legal ramifications and requirements governing the subject matter.

A computer forensics "how-to" for fighting malicious code and analyzing incidents. With our ever-increasing reliance on computers comes an ever-growing risk of malware. Security professionals will find plenty of solutions in this book to the problems posed by viruses, Trojan horses, worms, spyware, rootkits, adware, and other invasive software. Written by well-known malware experts, this guide reveals solutions to numerous problems and includes a DVD of custom programs and tools that illustrate the concepts, enhancing your skills. Security professionals face a constant battle against malicious software; this practical manual will improve your analytical capabilities and provide dozens of valuable and innovative solutions. Covers classifying malware, packing and unpacking, dynamic malware analysis, decoding and decrypting, rootkit detection, memory forensics, open source malware research, and much more. Includes generous amounts of source code in C, Python, and Perl to extend your favorite tools or build new ones, and custom programs on the DVD to demonstrate the solutions. Malware Analyst's Cookbook is indispensable to IT security administrators, incident responders, forensic analysts, and malware researchers.
Discover how the internals of malware work and how you can analyze and detect it. You will learn not only how to analyze and reverse malware, but also how to classify and categorize it, giving you insight into the intent of the malware. Malware Analysis and Detection Engineering is a one-stop guide to malware analysis that simplifies the topic by teaching you undocumented tricks used by analysts in the industry. You will be able to extend your expertise to analyze and reverse the challenges that malicious software throws at you. The book starts with an introduction to malware analysis and reverse engineering to provide insight on the different types of malware and also the terminology used in the anti-malware industry. You will know how to set up an isolated lab environment to safely execute and analyze malware. You will learn about malware packing, code injection, and process hollowing plus how to analyze, reverse, classify, and categorize malware using static and dynamic tools. You will be able to automate your malware analysis process by exploring detection tools to modify and trace malware programs, including sandboxes, IDS/IPS, anti-virus, and Windows binary instrumentation. The book provides comprehensive content in combination with hands-on exercises to help you dig into the details of malware dissection, giving you the confidence to tackle malware that enters your environment. What You Will Learn Analyze, dissect, reverse engineer, and classify malware Effectively handle malware with custom packers and compilers Unpack complex malware to locate vital malware components and decipher their intent Use various static and dynamic malware analysis
tools Leverage the internals of various detection engineering tools to improve your workflow Write Snort rules and learn to use them with Suricata IDS Who This Book Is For Security professionals, malware analysts, SOC analysts, incident responders, detection engineers, reverse engineers, and network security engineers "This book is a beast! If you're looking to master the ever-widening field of malware analysis, look no further. This is the definitive guide for you." Pedram Amini, CTO Inquest; Founder OpenRCE.org and ZeroDayInitiative
Unified Communications Forensics: Anatomy of Common UC Attacks is the first book to explain the issues and vulnerabilities and demonstrate the attacks, forensic artifacts, and countermeasures required to establish a secure (UC) environment. This book is written by leading UC experts Nicholas Grant and Joseph W. Shaw II and provides material never before found on the market, including: • analysis of forensic artifacts in common UC attacks • an in-depth look at established UC technologies and attack exploits • hands-on understanding of UC attack vectors and associated countermeasures • companion website http://secvoip.com giving readers access to the most up-to-date information on UC attacks. Provides key information for hackers and pen testers on the most current Unified Communications implementations The only book to explore and demonstrate how to work with digital artifacts from attacks within the UC environment Deals with UC security from multiple angles—less about theory and more about hands-on threat defense and forensics
Malware and rootkits are on the rise and becoming more complex, according to security company McAfee. Author speaks at major security conferences worldwide. Hands-on examples, attacks, and countermeasures are included in every chapter.

Copyright: 09e8adc69ff07185e8c511402044879a