SYNGRESS

THE BASICS

Technical edit by
Scott White

THE BASICS OF WEB HACKING

Tools and Techniques to Attack the Web

Josh Pauli

The Basics of Web Hacking

Tools and Techniques to Attack the Web

Josh Pauli

Scott White, Technical Editor

ELSEVIER

AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO

Syngress is an Imprint of Elsevier

Table of Contents

Chapter 8. Next Steps

Index

Copyright

Acquiring Editor: *Chris Katsaropoulos*
Editorial Project Manager: *Benjamin Rearick*
Project Manager: *Priya Kumaraguruparan*
Designer: *Mark Rogers*

Syngress is an imprint of Elsevier
225 Wyman Street, Waltham, MA 02451, USA

Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods or professional practices, may become necessary. Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information or methods described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

Library of Congress Cataloging-in-Publication Data

Pauli, Joshua J.
The basics of web hacking : tools and techniques to attack the Web / Josh Pauli.
pages cm
Includes bibliographical references and index.
ISBN 978-0-12-416600-4
1. Web sites–Security measures. 2. Web applications–Security measures. 3. Computer networks–Security measures. 4. Penetration testing (Computer security) 5. Computer hackers. 6. Computer crimes–Prevention. I. Title.
TK5105.59.P385 2013
005.8–dc23
2013017240

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

ISBN: 978-0-12-416600-4

Printed in the United States of America
13 14 15 10 9 8 7 6 5 4 3 2 1

For information on all Syngress publications, visit our website at www.syngress.com.

Dedication

This book is dedicated to my lovely wife, Samantha, and my two wonderful daughters, Liz and Maddie. I love you all very much.

Acknowledgments

Honey Bear

To my wife, Samantha: We've come a long way since being scared teenagers expecting a baby! You support no matter the projects I take on, your understanding no matter how much I complain, and yo composure no matter what comes at our family are legendary and have kept our family chuggi along.

Lizard

To my oldest daughter, Liz: Your work ethic, attention to detail, and drive to succeed are a inspiration to me. I'm looking forward to the coming years as you take on your next challenges, as have no doubt you will succeed with flying colors!

Baby Bird

To my youngest daughter, Maddie: Your smile and playful nature always pick me up and make m realize how good we have it. If four open-heart surgeries won't slow you down, what excuse do anybody else have? Keep smiling, playing, and being yourself—we're all better off that way!

Family and Friends

Huge thanks to Merm, Tara, Halverto, Stacy & Steph, Luke & Tracy, David, Dr. B, Crony, my DSU students, and everybody else that I've surely forgotten that have provided friendship and suppo *Salute!*

And a special note to Dr. Patrick Engebretson, a great friend and colleague, that I've shared man beers, fried goodies, stories, car rides, and office visits with. Your assistance through this publishin process has been a tremendous help. *Do work, big boy!*

Last, to my parents, Dr. Wayne and Dr. Crystal Pauli: It appears that those years of twisting my ea filling my mouth full of soap, and breaking wooden spoons on my butt have finally paid off! (Th stuff was allowed in the 1980s and it's obvious now that I wasn't the easiest child to raise.) Your lov and support have never wavered and I couldn't ask for better parents.

Security Community

Man, what a group. It doesn't matter if you're a complete beginner, a super l33t hacker, or anywhe in between, you're always welcome if you're willing to learn and explore. As a South Dakota guy, have my own personal "Mount Rushmore of Security": a group that not only is highly skilled security but also has provided me with a ton support.

- To Dr. Jared DeMott: You're one of the finest bug hunters/exploitation gurus in the world, but an even better family man and friend. With all your success it would be easy to forget about us "littl people" at Dakota State University, but instead you've never been a bigger supporter of our mission and goals.
- To Dave Kennedy: HUGS! You're one of the most encouraging security people that I've ever com across. The amount of fun you have working, training, speaking, and just hanging out with the security community is what this is all about. I'm glad our paths crossed and I look forward to many more years of watching you continue to flourish. MORE HUGS!
- To Eric Smith: I will never forget watching in awe as you dominated as a one-man red team for ou security competition at DSU. Your personal story of hard work, dedication, and hours spent perfecting your craft is one that I've relayed to my students hundreds of times. Thanks for always making time to come back to Madison, SD, and furthering your demigod status with our students
- To Dafydd Stuttard: I blame you for all of this! *The Web Application Hacker's Handbook* (*WAHH* that you authored with Marcus Pinto was one of the first premiere security books that I really dug into. After attending your classes, being the technical reviewer on the 2nd edition of *WAHH*, usin your Burp Suite web application hacking tool extensively, and exchanging countless e-mails with you, it's crystal clear that you're the Godfather of web application security. I've educated over 40 students with *WAHH* and Burp Suite and hope my book can serve as an on-ramp to your super highway.

Scott White—Technical Reviewer

A special thanks to Scott White for doing a tremendous job reviewing and cleaning up my work. Wit all the different directions you get pulled and requests for your time, I truly appreciate your expertis timeliness, and honest feedback. This book is much stronger because of your work!

Syngress Team

To all the fine folks at Syngress that took a chance on me and provided nothing but the best in servic feedback, and critiques in an uber-timely manner. Especially, Chris Katsaropoulos and Ben Rearick- your professionalism and tact are greatly appreciated and are the way an organization should operate

My Vices

In no particular order, I'd like to thank corndogs, Patron Silver, HOTEL32 at the Monte Carlo in La Vegas (especially @JohnnyLasVegas and Patty Sanchez), Mickey's malt liquor, fantasy football Pringles, and my 6-iron for helping me recharge.

Biography

Dr. Josh Pauli received his Ph.D. in software engineering from North Dakota State University (NDSU and now serves as an associate professor of cyber security at Dakota State University (DSU) Madison, SD. Dr. Pauli has published nearly 30 international journal and conference papers related software security and his work includes invited presentations from DEFCON, Black Hat, and Th National Security Agency. He teaches both undergraduate and graduate courses in software security DSU and is the program director for the DSU Cyber Corps. Dr. Pauli also conducts web applicatio penetration tests for an information security consulting firm. You can keep up with Josh on Twitter b following @CornDogGuy and visiting his DSU homepage at www.homepages.dsu.edu/paulij.

Foreword

The World Wide Web is a huge and expanding mass of application code. The majority of businesses governments, and other organizations are now on the web, exposing their systems and data to th world via custom application functionality. With today's development frameworks, it is easier tha ever to create a functional web application without knowing or doing anything about security. Wit today's technologies, that application is likely to be far more complex than those that have com before. Evolving technologies bring with them more attack surface and new types of attac Meanwhile, old vulnerabilities live on and are reintroduced into new applications by each generatio of coders.

In the recent past, numerous high-profile organizations have been compromised via their we applications. Though their PR departments may claim they were victims of highly sophisticate hackers, in reality the majority of these attacks have exploited simple vulnerabilities that have bee well understood for years. Smaller companies that don't feel under the spotlight may actually be eve more exposed. And many who are compromised never know about it.

Clearly, the subject of web application security is more critical today than ever before. There is significant need for more people to understand web application attacks, both on the offensive side (test existing applications for flaws) and on the defensive side (to develop more robust code in the fir place). If you're completely new to web hacking, this book will get you started. Assuming no existin knowledge, it will teach you the basic tools and techniques you need to find and exploit numerou vulnerabilities in today's applications. If your job is to build or defend web applications, it will ope your eyes to the attacks that your own applications are probably still vulnerable to and teach you ho to prevent them from happening.

Dafydd Stuttard

Creator of Burp Suite
Coauthor of The Web Application Hacker's Handbook

Introduction

Many of us rely on web applications for so many of our daily tasks, whether at work, at home, or
play, and we access them several times a day from our laptops, tablets, phones, and other devices. W
use these web applications to shop, bank, pay bills, attend online meetings, social network with frien
and family, and countless other tasks. The problem is that web applications aren't as secure as we
like to think, and most of the time the attacks used to gain access to a web application are relative
straightforward and simple. In fact, anyone can use widely available hacking tools to perform the
devastating web attacks.

This book will teach you how to hack web applications and what you can do to prevent the
attacks. It will walk you through the theory, tools, and techniques used to identify and exploit the mo
damaging web vulnerabilities present in current web applications. This means you will be able
make a web application perform actions it was never intended to perform, such as retrieve sensitiv
information from a database, bypass the login page, and assume the identity of other users. You'
learn how to select a target, how to perform an attack, what tools are needed and how to use them, a
how to protect against these attacks.

About This Book

This book is designed to teach you the fundamentals of web hacking from the ground up. It's for tho
of you interested in getting started with web hacking but haven't found a good resource. Basically,
you're a web hacking newbie, this is the book for you! This book assumes you have no previo
knowledge related to web hacking. Perhaps you have tinkered around with some of the tools, but yo
don't fully understand how or where they fit into the larger picture of web hacking.

Top web hacking experts have a firm grasp on programming, cryptography, bug huntin
exploitation development, database layout, data extraction, how network traffic works, and muc
more. If you don't have these skills, don't be discouraged! These knowledge and skills a
accumulated over the course of a career, and if you're just getting started with web hacking, yo
probably won't have all of these skills. This book will teach you the theory, tools, and techniqu
behind some of the most damaging web attacks present in modern web applications. You will gain no
only knowledge and skill but also confidence to transition to even more complex web hacking in th
future.

A Hands-On Approach

This book follows a very hands-on approach to introduce and demonstrate the content. Every chapt
will have foundational knowledge so that you know the *why* of the attack and detailed step-by-st
directions so that you know the *how* of the attack.

Our approach to web hacking has three specific targets: the web server, the web application, and th

web user. These targets all present different vulnerabilities, so we need to use different tools an techniques to exploit each of them. That's exactly what this book will do; each chapter will introduc different attacks that exploit these targets' vulnerabilities.

What's in This Book?

Each chapter covers the following material:

Chapter 1: The Basics of Web Hacking provides an overview of current web vulnerabilities and ho our hands-on approach takes aim at them.

Chapter 2: Web Server Hacking takes traditional network hacking methodologies and applies the directly to the web server to not only compromise those machines but also to provide a base knowledge to use in attacks against the web application and web user. Tools include Nmap, Nessu Nikto, and Metasploit.

Chapter 3: Web Application Recon and Scanning introduces tools, such as web proxies an scanning tools, which set the stage for you to exploit the targeted web application by finding existi vulnerabilities. Tools include Burp Suite (Spider and Intercept) and Zed Attack Proxy (ZAP).

Chapter 4: Web Application Exploitation with Injection covers the theory, tools, and technique used to exploit web applications with SQL injection, operating system command injection, and we shells. Tools include Burp Suite (specifically the functions and features of the Proxy Intercept an Repeater tools), sqlmap, John the Ripper (JtR), custom web shell files, and netcat.

Chapter 5: Web Application Exploitation with Broken Authentication and Path Traversal covers th theory, tools, and techniques used to exploit web applications with brute forcing logins, sessio attacks, and forceful browsing. Tools include Burp Suite (Intruder and Sequencer) and variou operating system commands for nefarious purposes.

Chapter 6: Web User Hacking covers the theory, tools, and techniques used to exploit other we users by exploiting web application cross-site scripting (XSS) and cross-site request forgery (CSR vulnerabilities as well as attacks that require no existing web server or web application vulnerabilitie but instead prey directly on the user's willingness to complete dangerous actions. The main tool choice will be Social-Engineer Toolkit (SET).

Chapter 7: Fixes covers the best practices available today to prevent all the attacks introduced in th book. Like most things security-related, the hard part is not identifying these mitigation strategies, b instead on how to best implement and test that they are doing what they are intended to do.

Chapter 8: Next Steps introduces where you can go after finishing this book to continue on yo hacking journey. There are tons of great information security groups and events to take part in. Som of you may want formal education, while others may want to know what certifications are especial applicable to this type of security work. A quick list of good books to consider is also provided.

A Quick Disclaimer

The goal of this book is to teach you how to penetrate web servers, web applications, and web user protect against common attacks; and generally improve your understanding of what web applicatio security is. In a perfect world, no one would use the tools and techniques discussed in this book in a unethical manner. But since that's not the case, keep the following in mind as you read along:

Think before you hack.

Don't do malicious things.

Don’t attack a target unless you have written permission.

Many of the tools and techniques discussed in this book are easily detected and traced.

If you do something illegal, you could be sued or thrown into jail. One basic assumption this boo
makes is that you understand right from wrong. Neither Syngress (this book’s publisher) nor I endors
using this book to do anything illegal. If you break into someone's server or web application witho
permission, don’t come crying to me when your local law enforcement agency kicks your door in!

CHAPTER 1

The Basics of Web Hacking

Chapter Rundown:

- What you need to know about web servers and the HTTP protocol
- The Basics of Web Hacking: our approach
- Common web vulnerabilities: they are still owning us
- Setting up a safe test environment so you don't go to jail

Introduction

There is a lot of ground to cover before you start to look at specific tools and how to configure ar execute them to best suit your desires to exploit web applications. This chapter covers all the are you need to be comfortable with before we get into these tools and techniques of web hacking. order to have the strong foundation you will need for many years of happy hacking, these are co fundamentals you need to fully understand and comprehend. These fundamentals include materi related to the most common vulnerabilities that continue to plague the web even though some of the have been around for what seems like forever. Some of the most damaging web applicatio vulnerabilities "in the wild" are still as widespread and just as damaging over 10 years after bein discovered.

It's also important to understand the time and place for appropriate and ethnical use of the tools ar techniques you will learn in the chapters that follow. As one of my friends and colleagues likes to sa about using hacking tools, "it's all fun and games until the FBI shows up!" This chapter includes step by-step guidance on preparing a sandbox (isolated environment) all of your own to provide a sa haven for your web hacking experiments.

As security moved more to the forefront of technology management, the overall security of o servers, networks, and services has greatly improved. This is in large part because of improve products such as firewalls and intrusion detection systems that secure the network layer. Howeve these devices do little to protect the web application and the data that are used by the web applicatio As a result, hackers shifted to attacking the web applications that directly interacted with all th internal systems, such as database servers, that were now being protected by firewalls and oth network devices.

In the past handful of years, more emphasis has been placed on secure software development and, a result, today's web applications are much more secure than previous versions. There has been strong push to include security earlier in the software development life cycle and to formalize th specification of security requirements in a standardized way. There has also been a huge increase the organization of several community groups dedicated to application security, such as the Open We Application Security Project. There are still blatantly vulnerable web applications in the wild, main because programmers are more concerned about functionality than security, but the days of easi exploiting seemingly *every* web application are over.

Therefore, because the security of the web application has also improved just like the network, th attack surface has again shifted; this time toward attacking web users. There is very little that netwo administrators and web programmers can do to protect web users against these user-on-user attac that are now so prevalent. Imagine a hacker's joy when he can now take aim on an unsuspecti technology-challenged user without having to worry about intrusion detection systems or we application logging and web application firewalls. Attackers are now focusing directly on the we

users and effectively bypassing any and all safeguards developed in the last 10 + years for networ and web applications.

However, there are still plenty of existing viable attacks directed at web servers and we applications in addition to the attacks targeting web users. This book will cover how all of the attacks exploit the targeted web server, web application, and web user. You will fully understand hov these attacks are conducted and what tools are needed to get the job done. Let's do this!

What Is a Web Application?

The term "web application" has different meanings to different people. Depending on whom you ta to and the context, different people will throw around terms like web application, web site, web-base system, web-based software or simply Web and all may have the same meaning. The widesprea adoption of web applications actually makes it hard to clearly differentiate them from previou generation web sites that did nothing but serve up static, noninteractive HTML pages. The term *we application* will be used throughout the book for any web-based software that performs actio (functionality) based on user input and usually interacts with backend systems. When a user interac with a web site to perform some action, such as logging in or shopping or banking, it's a we application.

Relying on web applications for virtually everything we do creates a huge attack surface (potenti entry points) for web hackers. Throw in the fact that web applications are custom coded by a huma programmer, thus increasing the likelihood of errors because despite the best of intentions. Humar get bored, hungry, tired, hung-over, or otherwise distracted and that can introduce bugs into the we application being developed. This is a perfect storm for hackers to exploit these web applications th we rely on so heavily.

One might assume that a web application vulnerability is merely a human error that can be quick fixed by a programmer. Nothing could be further from the truth: most vulnerabilities aren't easi fixed because many web application flaws dates back to early phases of the software developme lifecycle. In an effort to spare you the gory details of software engineering methodologies, just reali that security is much easier to deal with (and much more cost effective) when considered initially the planning and requirements phases of software development. Security should continue as a drivi force of the project all the way through design, construction, implementation, and testing.

But alas, security is often treated as an afterthought too much of the time; this type of developme leaves the freshly created web applications ripe with vulnerabilities that can be identified an exploited for a hacker's own nefarious reasons.

What You Need to Know About Web Servers

A web server is just a piece of software running on the operating system of a server that allow connections to access a web application. The most common web servers are Internet Informatio Services (IIS) on a Windows server and Apache Hypertext Transfer Protocol (HTTP) Server on Linux server. These servers have normal directory structures like any other computer, and it's thes directories that house the web application.

If you follow the Windows **next, next, next, finish** approach to installing an IIS web server, yo will end up with the default *C:\Inetpub\wwwroot* directory structure where each application will hav its own directories within *wwwroot* and all vital web application resources are contained within it.

Linux is more varied in the file structure, but most web applications are housed in the */var/ww* directory. There are several other directories on a Linux web server that are especially relevant to we hacking:

- */etc/shadow*: This is where the password hashes for all users of the system reside. This is the "key to the kingdom"!
- */usr/lib*: This directory includes object files and internal binaries that are not intended to be executed by users or shell scripts. All dependency data used by the application will also reside in this directory. Although there is nothing executable here, you can really ruin somebody's day by deleting all of the dependency files for an application.
- */var/**: This directory includes the files for databases, system logs, and the source code for web application itself!
- */bin*: This directory contains programs that the system needs to operate, such as the shells, **ls**, **gre** and other essential and important binaries. *bin* is short for binary. Most standard operating systen commands are located here as separate executable binary files.

The web server is a target for attacks itself because it offers open ports and access to potential vulnerable versions of web server software installed, vulnerable versions of other software installe and misconfigurations of the operating system that it's running on.

What You Need to Know About HTTP

The HTTP is the agreed upon process to interact and communicate with a web application. It completely plaintext protocol, so there is no assumption of security or privacy when using HTT HTTP is actually a stateless protocol, so every client request and web application response is a bran new, independent event without knowledge of any previous requests. However, it's critical that th web application keeps track of client requests so you can complete multistep transactions, such online shopping where you add items to your shopping cart, select a shipping method, and ent payment information.

HTTP without the use of cookies would require you to relogin during each of those steps. That just not realistic, so the concept of a session was created where the application keeps track of yo requests after you login. Although sessions are a great way to increase the user-friendliness of a we application, they also provide another attack vector for web applications. HTTP was not original created to handle the type of web transactions that requires a high degree of security and privacy. Yo can inspect all the gory details of how HTTP operates with tools such as Wireshark or any local HTT proxy.

The usage of secure HTTP (HTTPS) does little to stop the types of attacks that will be covered this book. HTTPS is achieved when HTTP is layered on top of the Secure Socket Layer/Transpo Layer Security (SSL/TLS) protocol, which adds the TLS of SSL/TLS to normal HTTP request an responses. It is best suited for ensuring man-in-the-middle and other eavesdropping attacks are n successful; it ensures a "private call" between your browser and the web application as opposed having a conversation in a crowded room where anybody can hear your secrets. However, in our usag HTTPS just means we are going to be communicating with the web application over an encrypte communication channel to make it a private conversation. The bidirectional encryption of HTTPS wi not stop our attacks from being processed by the waiting web application.

HTTP Cycles

One of the most important fundamental operations of every web application is the cycle of reques made by clients' browsers and the responses returned by the web server. It's a very simple premis that happens many of times every day. A browser sends a request filled with parameters (variables holding user input and the web server sends a response that is dictated by the submitted request. Th web application may act based on the values of the parameters, so they are prime targets for hackers attack with malicious parameter values to exploit the web application and web server.

Noteworthy HTTP Headers

Each HTTP cycle also includes headers in both the client request and the server response that transm details about the request or response. There are several of these headers, but we are only concerne with a few that are most applicable to our approach covered in this book.

The headers that we are concerned about that are set by the web server and sent to the client browser as part of the response cycle are:

- *Set-Cookie*: This header most commonly provides the session identifier (cookie) to the client to ensure the user's session stays current. If a hacker can steal a user's session (by leveraging attack covered in later chapters), they can assume the identity of the exploited user within the application.
- *Content-Length*: This header's value is the length of the response body in bytes. This header is helpful to hackers because you can look for variation in the number of bytes of the response to help decipher the application's response to input. This is especially applicable when conducting brute force (repetitive guessing) attacks.
- *Location*: This header is used when an application redirects a user to a new page. This is helpful t a hacker because it can be used to help identify pages that are only allowed after successfully authenticating to the application, for example.

The headers that you should know more about that are sent by the client's browser as part of th web request are:

- *Cookie*: This header sends the cookie (or several cookies) back to the server to maintain the user's session. This cookie header value should always match the value of the set-cookie header that wa issued by the server. This header is helpful to hackers because it may provide a valid session with the application that can be used in attacks against other application users. Other cookies are not a juicy, such as a cookie that sets your desired language as English.
- *Referrer*: This header lists the webpage that the user was previously on when the next web reques was made. Think of this header as storing the "*the last page visited*." This is helpful to hackers because this value can be easily changed. Thus, if the application is relying on this header for any sense of security, it can easily be bypassed with a forged value.

Noteworthy HTTP Status Codes

As web server responses are received by your browser, they will include a status code to signal wh type of response it is. There are over 50 numerical HTTP response codes grouped into five famili that provide similar type of status codes. Knowing what each type of response family represen allows you to gain an understanding of how your input was processed by the application.

- **100s**: These responses are purely informational from the web server and usually mean that

additional responses from the web server are forthcoming. These are rarely seen in modern web server responses and are usually followed close after with another type of response introduced below.

- **200s**: These responses signal the client's request was successfully accepted and processed by the web server and the response has been sent back to your browser. The most common HTTP status code is **200 OK**.
- **300s**: These responses are used to signal redirection where additional responses will be sent to the client. The most common implementation of this is to redirect a user's browser to a secure homepage after successfully authenticating to the web application. This would actually be a **302 Redirect** to send another response that would be delivered with a **200 OK**.
- **400s**: These responses are used to signal an error in the request from the client. This means the user has sent a request that can't be processed by the web application, thus one of these common status codes is returned: **401 Unauthorized**, **403 Forbidden**, and **404 Not Found**.
- **500s**: These responses are used to signal an error on the server side. The most common status codes used in this family are the **500 Internal Server Error** and **503 Service Unavailable**.

Full details on all of the HTTP status codes can be reviewed in greater detail http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html.

The Basics of Web Hacking: Our Approach

Our approach is made up of four phases that cover all the necessary tasks during an attack.

1. Reconnaissance
2. Scanning
3. Exploitation
4. Fix

It's appropriate to introduce and discuss how these vulnerabilities and attacks can be mitigated, th there is a fix phase to our approach. As a penetration tester or ethical hacker, you will get sever questions after the fact related to how the discovered vulnerabilities can be fixed. Consider t inclusion of the fix phase to be a resource to help answer those questions.

Our Targets

Our approach targets three separate, yet related attack vectors: the web server, the web applicatio and the web user. For the purpose of this book, we will define each of these attack vectors as follows:

1. *Web server*: the application running on an operating system that is hosting the web applicatio We are NOT talking about traditional computer hardware here, but rather the services running on open ports that allow a web application to be reached by users' internet browsers. The web server may be vulnerable to network hacking attempts targeting these services in order to gai unauthorized access to the web server's file structure and system files.
2. *Web application*: the actual source code running on the web server that provides the functionality that web users interact with is the most popular target for web hackers. The web application may be susceptible to a vast collection of attacks that attempt to perform unauthorized actions within the web application.
3. *Web user*: the internal users that manage the web application (administrators and programmer and the external users (human clients or customers) of the web applications are worthy target

of attacks. This is where a cross-site scripting (XSS) or cross-site request forgery (CSRF) vulnerabilities in the web application rear their ugly heads. Technical social engineering attacks that target web users and rely on no existing web application vulnerabilities are also applicable here.

The vulnerabilities, exploits, and payloads are unique for each of these targets, so unique tools an techniques are needed to efficiently attack each of them.

Our Tools

For every tool used in this book, there are probably five other tools that can do the same job. (Th same goes for methods, too.) We'll emphasize the tools that are the most applicable to beginner we hackers. We recommend these tools not because they're easy for beginners to use, but because they'r fundamental tools that virtually every professional penetration tester uses on a regular basis. It paramount that you learn to use them from the very first day. Some of the tools that we'll be usin include:

- *Burp Suite*, which includes a host of top-notch web hacking tools, is a must-have for any web hacker and it's widely accepted as the #1 web hacking tool collection.
- *Zed Attack Proxy* (*ZAP*) is similar to Burp Suite, but also includes a free vulnerability scanner that's applicable to web applications.
- Network hacking tools such as *Nmap* for port scanning, *Nessus* and *Nikto* for vulnerability scanning, and *Metasploit* for exploitation of the web server.
- And other tools that fill a specific role such as *sqlmap* for SQL injection, *John the Ripper* (*JtR*) fo offline password cracking, and the *Social Engineering Toolkit* (*SET*) for technical social engineering attacks against web users!

Web Apps Touch Every Part of IT

Another exciting tidbit for web hackers is the fact that web applications interact with virtually eve core system in a company's infrastructure. It's commonplace to think that the web application is ju some code running on a web server safely tucked away in an external DMZ incapable of doing seriou internal damage to a company. There are several additional areas of a traditional IT infrastructure th need to be considered in order to fully target a system for attack, because a web application's reach much wider than the code written by a programmer. The following components also need to b considered as possible attack vectors:

- Database server and database: the system that is hosting the database that the web application use may be vulnerable to attacks that allow sensitive data to be created, read, updated, or deleted (CRUD).
- File server: the system, often times a mapped drive on a web server, that allows file upload and/o download functionality may be vulnerable to attacks that allow server resources to be accessed from an unauthorized attacker.
- Third-party, off-the-shelf components: modules of code, such as content management systems (CMSs), are a definitely a target because of the widespread adoption and available documentation of these systems.

Existing Methodologies

Several attack methodologies provide the processes, steps, tools, and techniques that are deemed to b best practices. If you're a white hat hacker, such activities are called penetration testing (*pen test* f short or *PT* for even shorter), but we all realize they are the same activities as black hat hacking. Th two most widely accepted pen test methodologies today are the *Open-Source Security Testin Methodology Manual* (*OSSTM*) and the *Penetration Testing Execution Standard* (*PTES*).

The Open-Source Security Testing Methodology Manual (OSSTM)

The OSSTM was created in a peer review process that created cases that test five sections:

1. Information and data controls
2. Personnel security awareness levels
3. Fraud and social engineering levels
4. Computer and telecommunications networks, wireless devices, and mobile devices
5. Physical security access controls, security process, and physical locations

The OSSTM measures the technical details of each of these areas and provides guidance on what do before, during, and after a security assessment. More information on the OSSTM can be found the project homepage at http://www.isecom.org/research/osstmm.html.

Penetration Testing Execution Standard (PTES)

The new kid on the block is definitely the PTES, which is a new standard aimed at providing commo language for all penetration testers and security assessment professionals to follow. PTES provides client with a baseline of their own security posture, so they are in a better position to make sense