Ethical Hacking and Attack Tools

Kenneth Ingham

September 29, 2009

1 Course overview

Attackers have at their disposal a large collection of tools that aid their exploiting systems. If you plan to defend against attacks, knowledge of these tools and the techniques behind their use is imperative. This class covers vulnerabilities in systems, how attackers locate these security holes, and how they can then exploit them to achieve their goals. Additionally, the class covers defenses against the attacker's tools and techniques. Labs in this course are of two types: (1) Attacking a vulnerable system, and (2) preventing your classmates from successfully attacking your system.

2 Course objectives

- Learn techniques attackers use to compromise systems.
- Learn how to defend against these attacks.
- Learn many of the tools available to attackers.

3 Student background

If you are attending this class, then we assume that

- You have a basic knowledge of Windows, including the ability to install software.
- You are familiar with the Linux command line, including using a text editor, file manipulation commands, and basic system administration tasks.
- You can read documentation for both of these systems.
- You know the ISO seven-layer model for networking, including what services each layer provides.

- You understand the basics of TCP, UDP, and ICMP and know the differences between the protocols.
- You understand programming in a major programming language (e.g., C, C++, C#, Java, PHP, Python, Ruby, ...).

4 Logistics

The class lasts 4 days. The student computers need to run both Linux and Windows. Notable amounts of class time will be saved by having the tools pre-installed on the class machines. The total time for the students to install all the software exceeds two hours, and this time is better spent on class topics.

Additionally, the class needs the Backtrack 2 attack tools CD. The class uses the following software:

- Internet access
- A collection of virtual and/or real machines for the class to scan (The instructor must set these up. A honeynet configuration is on the class web site. Other virtual machines are the responsibility of the instructor.)
- A web browser
- Backtrack 2 CD
- Firefox (on Linux distribution but needed for Windows)
- Firefox Web Developer toolbar
- Gnu C/C++ compiler (on Linux distribution but needed for Windows)
- Internet access
- Java JDK 1.6
- Metasploit framework (on class web site)
- Missing from blackbox-attack.tex
- Missing from reverse-engr.tex
- Netcat (on Linux distribution)
- Nikto (on class web site)
- One or more (virtual and/or real) machines supporting SNMP on the class network
- Paros proxy
- PortSwigger burp suite
- Scapy (on class web server)
- WebGoat v5.2
- WebScarab
- *WinARPWatch* (on class web site)
- *arping* (on Linux distribution)
- *arpwatch* (on Linux distribution)
- *dig* (on Linux distribution)
- *dsniff* (on Linux distribution and class web site)
- *emacs* (on Linux distribution)

- *etherflood* (on class web site)
- *ettercap* (on class web site)
- *fping* (on Linux distribution)
- *fudp* (on class web server)
- *gcc* (on Linux distribution)
- *gdb* (on Linux distribution but needed for Windows)
- *gdb* (on Linux distribution)
- *host* (on Linux distribution)
- *hping* (on class web server)
- *hunt* (on class web server)
- *juggernaut* (on class web server
- *nessus* (on class web site)
- *netcat* (get this for Windows as part of Cygwin from cygwin.com; On Linux it is on the distribution media)
- *nmap* (on Linux distribution)
- *nm* (on Linux distribution)
- *objdump* (on Linux distribution but needed for Windows)
- *p0f* (on class web site)
- *pscan*
- *rats* (on class web site)
- *rpcinfo* (on Linux distribution)
- *snmpcheck* (on class web site)
- *snmpget* (on Linux distribution)
- *snmpset* (on Linux distribution)
- *snmpwalk* (on Linux distribution)
- *stresser* (on class web server)
- *tcpdump* (on Linux distribution)
- *tcptraceroute* (on class web server)
- *telnet* (on Linux and Windows distributions)
- *telnet* (on Linux distribution)
- *traceroute* (on Linux distribution)
- *tracert* (on Windows distribution)
- *trout* (on class web server)
- *udpflood.exe* (on class web server)
- *whois* (on Linux distribution)
- *winfingerprint* (on class web site)
- *wireshark* (on Linux distribution but needed for Windows)
- *xprobe2* (on class web site)
- a class web server that can run perl CGI programs
- at least one of *ddd* or the GNU Visual Debugger (on Linux distribution)
- development tools such as *make* (on Linux distribution but needed for Windows)
- development tools such as *make* (on Linux distribution)

- perl 5.8 (On Linux distribution but needed for Windows)
- sufficient network address space for a honeynet (private IP addresses are acceptable)

Because the students will be using a collection of attack tools, we **strongly** recommend that the class network be isolated from the corporate network. However, Internet access (HTTP, HTTPS, DNS, possibly POP or IMAP) is necessary. This access can be through a firewall, but the firewall cannot prevent access to "hacker sites".

The students will be gathering publically-available information about the customer's company. Permission to gather this information from remote sites should be explicitly given. Attackers are gathering this information already. However, the gathering of this information may reveal security issues in the corporate presence on the Internet. Students will be explicitly directed not to attack production machines.

The class network needs to be a /16. Private Internet space is acceptable.

Students will also need an external email address. Web-based email is preferred, but IMAP and/or POP access will also work. Note that the mail user agent (web or other) must allow the user to view the full set of headers. For an extra fee, this access can be provided.

In addition to the machines the students will use, the instructor needs a machine with at least a dual-core 64-bit CPU with 4GB of RAM and a DVD reader. More memory would be an asset, as would additional CPUs. This machine will run many of the victim systems as virtual machines, as well as simulate a victim network full of machines. Software for this machine will be provided on a DVD, and the instructor will install and test the setup on the day before the class starts.

Additionally, for an extra fee we can supply a bridging firewall placed between the class network and the outside world that will enforce the rules and guidelines, as well as log any attempted violations of the policy. To utilize this service requires:

- the class has its own Ethernet switch, and
- the instructor can access the room containing the switch and can place the firewall between the class switch and the outside world.
- The location for the firewall must provide sufficient cooling (ambient temperature not to exceed 75F), power (500 watts), and a stable physical location for the firewall machine.

Using this firewall is recommended because it will also be properly configured to act as a local DHCP and DNS server as well as a web proxy and provide network address translation (NAT) services. The instructor will set up the firewall the day before the class starts.

The class needs a web server for the class web site. The instructor's laptop may be this web server; otherwise the machine provided in the classroom for the instructor is a good choice. This machine obviously will need web server software installed.

5 Class outline

1. Introduction (Lecture: 15; Lab: 0)

(a) Class Introductions
(b) Class Logistics
    i. Class schedule
    ii. Breaks
    iii. Question policy
    iv. Break room and restroom locations
    v. Assumptions about your background
(c) Typographic conventions
(d) What the class covers

2. Ethical hacking introduction (Lecture: 45; Lab: 15)
    (a) Introduction
    (b) Vocabulary
    (c) Attack goals
    (d) Types of vulnerabilities that exist
    (e) Attack procedure
    (f) Defending
    (g) Risk analysis
    (h) Legal issues
    (i) Notes about this class
    (j) Summary
    (k) Lab
3. Intelligence about the target (Lecture: 50; Lab: 35)
    (a) Introduction
    (b) Internal and external web sites
    (c) Network information
        i. Finding this information
        ii. *whois*
        iii. Simple Forward and Reverse DNS Lookups
    (d) Netcraft
    (e) Email
    (f) Web tools
        i. Maltego
    (g) Defenses
    (h) Summary
    (i) Lab
4. Network mapping (Lecture: 40; Lab: 35)
    (a) Introduction
    (b) Scanning for hosts
        i. ARP pings
        ii. *arping*

iii. *nmap*
iv. ICMP scans
A. *nmap*
B. *fping*
v. TCP
A. *nmap*
B. *hping*

(c) *traceroute*
i. *tcptraceroute*
ii. *mtr*
iii. *trout*

(d) Firewalls
(e) Defenses
(f) Lab

5. Host mapping (Lecture: 45; Lab: 30)

(a) Introduction
(b) Port scanning
i. TCP
A. TCP scan types
ii. TCP scanning with *nmap*
iii. TCP scanning with *netcat* or *nc*
iv. UDP

(c) Remote procedure call protocols
(d) Banner grabbing
i. Tools
A. *telnet*
B. *netcat*
C. *nmap*

(e) Protocol identification with *amap*
(f) OS identification
i. Active
ii. Passive OS identification

(g) Evading detection
(h) Defenses
(i) Lab

6. SNMP mapping (Lecture: 25; Lab: 30)

(a) Introduction
i. SNMP traps
ii. SNMP Overview

(b) What you can do with SNMP
(c) SNMP Versions and their security (or lack thereof)
(d) Tools
    i. *snmpwalk*
    ii. *snmpget*
    iii. *snmpset*
    iv. *onesixtyone*
    v. *snmpcheck*
(e) Defenses
(f) Lab

7. Network Monitoring and Eavesdropping (Lecture: 60; Lab: 75)
(a) Introduction
(b) Monitoring tools
    i. *tcpdump*
    ii. *wireshark*
    iii. *webspy*
(c) Password grabbers
    i. *dsniff*
(d) Attacking switches
    i. *macof*
    ii. *etherflood*
    iii. *arpspoof*
    iv. *ettercap*
(e) DNS spoofing
    i. *dnsspoof*
(f) What to look for or do when performing these attacks
    i. *webmitm*
    ii. *sshmitm*
(g) Defenses
(h) Lab

8. Attacking the network protocols (Lecture: 25; Lab: 35)
(a) SYN flooding
    i. Launching a SYN flood attack
(b) TCP hijacking
(c) TCP session termination
(d) Port flooding
(e) Defenses
(f) Lab

9. Network traffic injection (Lecture: 25; Lab: 50)

(a) Why inject traffic
(b) Scapy
(c) Hping
(d) *packETH*
(e) Defenses
(f) Lab

10. Reverse Engineering (Lecture: 15; Lab: 45)
    (a) Introduction
    (b) IDA Pro
    (c) Linux tools
        i. Static analysis
            A. *strings*
            B. *nm*
            C. *objdump*
        ii. Dynamic analysis
            A. gdb
            B. strace
            C. ltrace
    (d) *wine*
    (e) Java
    (f) Microsoft CLR
    (g) Lab
11. Black-box Testing (Lecture: 30; Lab: 35)
    (a) Introduction
    (b) Syntax Testing
    (c) Equivalence Partitioning
    (d) Boundary Condition Testing
    (e) Fuzzing
    (f) Resource Limits and Failure Conditions
    (g) Tools for Testing
        i. Fuzz testing tools
        ii. The Peach Fuzzer
    (h) Summary
    (i) Lab
12. How HTTP works (Lecture: 20; Lab: 25)
    (a) Introduction
    (b) Common Gateway Interface (CGI) Parameters
    (c) GET and POST
    (d) Cookies
    (e) Lab

13. Attacking Web Applications (Lecture: 50; Lab: 45)
    (a) Introduction
    (b) PortSwigger's Burp Suite
    (c) WebScarab
    (d) Paros Suite
    (e) Nikto
    (f) Firefox Web Developer toolbar
    (g) WebGoat
    (h) Summary
    (i) Lab
14. State and the web (Lecture: 25; Lab: 65)
    (a) Overview
    (b) Ways of tracking state
        i. Hidden fields in forms
        ii. Cookies
        iii. CGI parameters
        iv. HTTP Referer field
    (c) Session hijacking
        i. Solutions
    (d) Summary
    (e) Lab
15. Other Injection attacks (Lecture: 15; Lab: 35)
    (a) Introduction
    (b) SQL injection
        i. Overview
        ii. Solutions
    (c) Shell code injection
        i. Overview
        ii. Solutions
    (d) Summary
    (e) Lab
16. Cross-site scripting (XSS) (Lecture: 40; Lab: 30)
    (a) Overview
    (b) A simple example
    (c) Example XSS attacks
        i. DoS the user's browser
            A. Session hijacking
        ii. DoS a web server
        iii. Make a web site contents not what the owner expects

        iv. Port scanning
        v. Worms and viruses
    (d) Locations to place script references
    (e) Ways attackers try to obscure XSS
    (f) Types of XSS attacks
    (g) XSS is not just for HTML
    (h) XSS solutions
    (i) Summary
    (j) Lab
17. Buffer overflow introduction (Lecture: 20; Lab: 0)
    (a) Introduction
    (b) A simple buffer overflow example
    (c) Memory layout
        i. Example
    (d) Summary
18. Stack overflows (Lecture: 20; Lab: 60)
    (a) Introduction
    (b) Exploit: calling another function
    (c) Other exploits for stack overflows
    (d) More stack overflow information
    (e) A real stack overflow exploit program
    (f) Avoiding these problems
    (g) Summary
    (h) Lab
19. Format string errors (Lecture: 25; Lab: 60)
    (a) Introduction
        i. Example
    (b) It gets worse!
    (c) A real attack program
    (d) The problem
    (e) How can you avoid this problem?
    (f) Summary
    (g) Lab
20. Pointer issues (Lecture: 35; Lab: 50)
    (a) Introduction
        i. Example
    (b) Pointers to functions
    (c) C++ virtual method table
        i. Example

ii. Destructors

(d) Global offset table (GOT)

(e) The `.dtors` section

i. Example

(f) Exit handlers

i. Example

(g) **setjmp/longjmp**

(h) Exception handling

(i) Avoiding these problems

(j) Summary

(k) Lab

21. Vulnerability analysis (Lecture: 45; Lab: 60)

(a) Introduction

(b) Vulnerability databases

(c) Attacker web sites

(d) Vulnerability assessment tools

(e) Other non-commercial penetration testing tools

(f) Commercial penetration testing tools

(g) *nessus*

i. Installing and running *nessus*

(h) Defenses

(i) Lab

22. Metasploit (Lecture: 40; Lab: 45)

(a) Introduction

(b) Using metasploit

i. Web interface

ii. Console interface

iii. Command-line interface

(c) Auxiliary modules

(d) The Data Store

(e) Summary

(f) Lab

Appendices

A. Cryptography Overview (Lecture: 70; Lab: 55)

(a) Introduction

i. Cryptographic Applications

ii. Open design

(b) Cryptographic Primitives

i. Cryptographic hash functions
ii. Symmetric key encryption
iii. Public key encryption

(c) Digital signatures
(d) Public Key Management
i. The Problem
ii. Certificates
iii. Trust Models
iv. Example: PGP/GnuPG
v. Example: SSL/TLS
vi. Overview
A. The server
B. The client

(e) Random numbers
(f) Parameter sizes
(g) Insecure Cryptography
i. Key management errors

(h) Do not innovate in cryptography
(i) Summary
(j) Lab

B. Debugging with gdb (Lecture: 25; Lab: 30)

(a) Introduction
(b) Compiling programs to be debugged
(c) Working in *gdb*
i. Starting and exiting *gdb*
ii. *gdb* commands
iii. Help
iv. Info
v. Show
vi. Running programs

(d) Breakpoints and watchpoints
(e) Continuing and single stepping
(f) Viewing data and the stack
(g) Demonstration
(h) GUI front ends for *gdb*
(i) Lab