Offensive Software Exploitation

Summer 2020

Ali Hadi
@binaryz0ne

Vulnerability Identification

a quick road to bug hunting …

Part #1

Outline – Bug Hunting

- Bug Hunting
- 4 Fun & Profit
- Taking Advantages of Bugs
- Exploits Language
- Bug Hunting Formal Process
- Common Techniques

Wait …

- Before we proceed into exploitation, do you know what we mean by a:
  - "Vulnerability" or "Security hole" ?

- *INFOSEC 101 …. ☺*

Bug Hunting

- "Bug hunting is the process of finding bugs in software or hardware" [1]

- Security bugs (aka software security vulnerabilities and security holes) allows attackers to:
  - Remotely compromise systems
  - Escalate local privileges
  - Cross privilege boundaries
  - Wreak havoc on a system!

4 Fun & Profit

- Finding security bugs was done for fun and to get media attention
- Today, organizations are paying for security researchers to identify bugs
  - Bounty programs (Google, FaceBook, Twitter, RedHat, etc)
  - Zero Day Initiative (ZDI)
  - iDefense
  - Tipping Point
  - Pwn2Own
  - Others? Please add

Taking Advantages of Bugs

- Software that take the advantages of a software vulnerability are called “exploits”
- Exploiting a widely used application, OS, protocol, etc … will lead to huge media attention and coverage
  - Road to become a Hacking Star ☺

Exploits Language

- No specific language for writing exploits
- Exploits can be written using any programming language
  - C, C++, Perl, JavaScript, Assembly, and Python!
- I prefer Python for its simplicity and for the huge range of libraries that could be used for creating a PoC or a working exploit

Bug Hunting Formal Process

- Writing software is a *human art*, and two different coders may code the same function with the same requirements *differently*!

- For that reason IMHO, Bug Hunting is a *human art* too!

- No formal process to finding bugs in SW, but there are a couple of techniques that can be used for bug discovery

Common Techniques

- Static Analysis
  - Static Code Analysis
  - Reverse Engineering
- Dynamic Analysis
  - Debugging
  - Fuzzing
- Each technique has its pros and cons
  - Bug hunters mix it up

Static Analysis

- Static Code Analysis
  - Code is needed
  - Tedious and time consuming
  - Requires high knowledge and/or skills with given language
  - Costs a lot (expensive)

- Reverse Engineering
  - Code not needed
  - Requires the binary file
  - Time consuming
  - High technical skill is needed (assembly!)

Dynamic Analysis

- Will be covered while we progress through the course

Static Analysis and RE are out of the scope of this course…

General Bug Hunting Methodology

Understand the Application

- Read specs / documentation
  - understand purpose or business logic

- Examine attack surface
  - inputs, configuration

- Identify target components an attacker would hit
  - think like an attacker to defend better:
    - try to hit the Database for SQLi?
    - try to upload a file?
    - try to spawn a shell?

What Leads to Bugs?

- Miscalculations
- Failure to validate input
- Programmer failure to understand an API
- Failure to validate results: operations, functions, etc
- Application state failures
- Complex protocols
- Complex file formats
- Complex encoding / decoding / expansion
- etc

References

- A Bug Hunter’s Diary, Tobias Klein, No Starch Press
- Sam Bowne, Malware Analysis Course Slides, http://samsclass.info/126/126_F13.shtml
- Fuzz Testing, http://en.wikipedia.org/wiki/Fuzz_testing
- Fuzzing: Brute Force Vulnerability Discovery, Michael Sutton, et al, Addison-Wesely
- University of Wisconsin Fuzz Testing (the original fuzz project)
- Fuzzing 101, NYU/Poly.edu, Mike Zusman, http://pentest.cryptocity.net/fuzzing/
- Fuzzing for Security Flaws, John Heasman, Stanford University
- EVERYONE HAS HIS OR HER OWN FUZZER, BEIST (BEISTLAB/GRAYHASH), www.codeengn.com
- An Introduction to SPIKE, the Fuzzer Creation Kit, Dave Aitel, http://www.docstoc.com/docs/2687423/An-Introduction-to-SPIKE-the-Fuzzer-Creation-Kit---PowerPoint
- Common Vulnerablities and Exposures, http://cve.mitre.org/
- Common Weakness Enumeration, http://cwe.mitre.org/
- Seven kingdoms of weaknesses Taxonomy, http://cwe.mitre.org/documents/sources/SevenPerniciousKingdomsTaxonomyGraphic.pdf
- Common Configuration Enumeration, http://cce.mitre.org/
- National Vulnerability Database, http://nvd.nist.gov/home.cfm
- Exploit Database, http://exploit-db,com
- http://www.security-database.com/toolswatch/+-Fuzzers-+.html
- http://caca.zoy.org/wiki/zzuf
- https://code.google.com/p/ouspg/wiki/Radamsa