Introduction to Security Vulnerabilities

Ming Chow (mchow@cs.tufts.edu)

Twitter: @0xmchow

Learning Objectives

- By the end of this week, you will be able to:
 - Be exposed to and understand the difference between CVE and CWE

Why Talk About Vulnerabilities Now?

- Terminology and vocabulary

- The issue of vulnerability disclosure is an ongoing debate, one of the really hard problems in Cyber Security

- Understand why software development is very difficult; all software have bugs but some bugs are worse than others

- Cryptography and Web Security have a lot to do with vulnerabilities

Motivation: How Hard is Software Development?

- Here's a problem for you: *on paper*, write a binary search program.
  - Recall binary search: find the position of a target value within a sorted list by "comparing the target value to the middle element of the array; if they are unequal, the half in which the target cannot lie is eliminated and the search continues on the remaining half until it is successful or the remaining half is empty." (Wikipedia)
- A humbling exercise, pitfalls galore including off-by-one errors: https://stackoverflow.com/questions/504335/what-are-the-pitfalls-in-implementing-binary-search
- Thank Jon Bentley via Matt Blaze for this https://twitter.com/mattblaze/status/771086675258802176

From Matt Blaze and Sandy Clark's talk
"Crypto War II: Updates from the Trenches"
at The Eleventh HOPE Conference in NYC,
July 2016
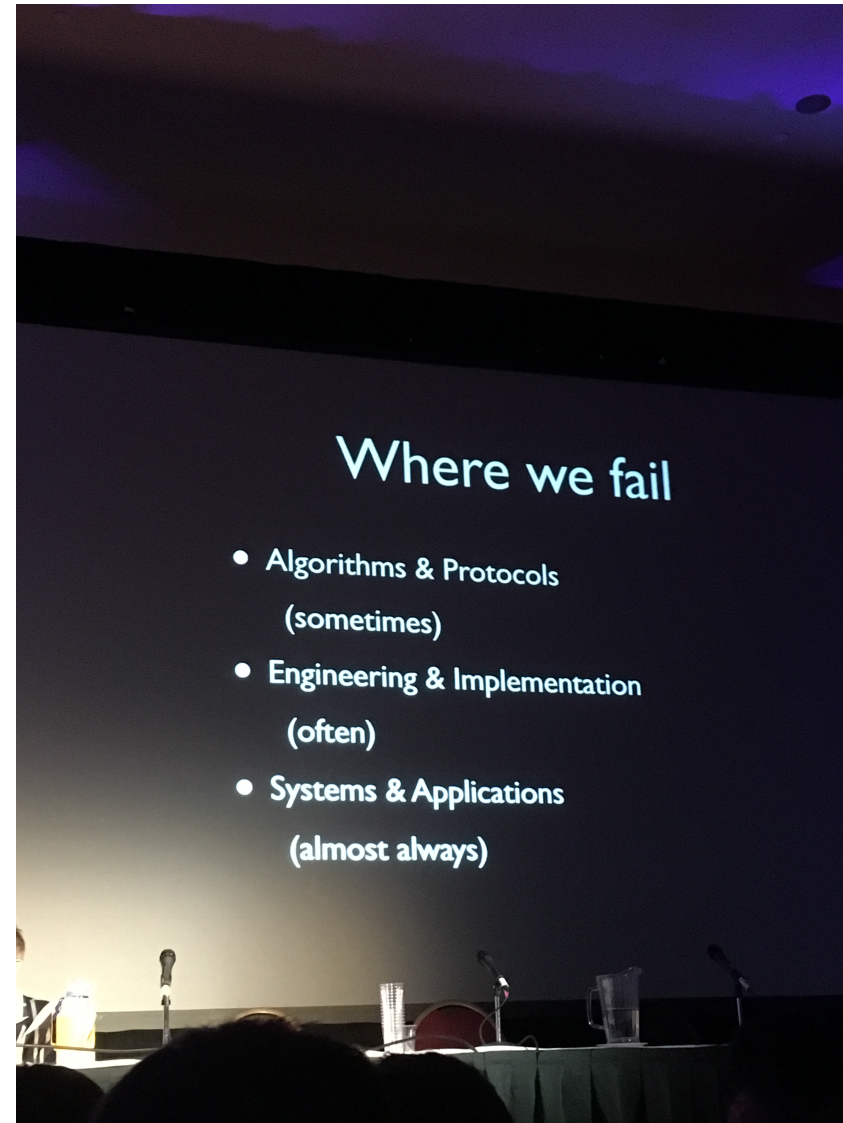
Vocabulary: Bug vs Flaw

- **Bug** - An error that exists in the implementation-level (i.e. only exist in source code); very correctable

- **Flaw** - An error at a much deeper level, particularly in the design, and likely in the code level; can be very difficult and costly to correct

Vocabulary: Vulnerability

- "A weakness in the computational logic (e.g., code) found in software and some hardware components (e.g., firmware) that, when exploited, results in a negative impact to confidentiality, integrity, OR availability."

- Source: https://cve.mitre.org/about/terminology.html

- Furthermore: *"Examples of vulnerabilities include:*
  - *phf (remote command execution as user "nobody")*
  - *rpc.ttdbserverd (remote command execution as root)*
  - *world-writeable password file (modification of system-critical data)*
  - *default password (remote command execution or other access)*
  - *denial of service problems that allow an attacker to cause a Blue Screen of Death*
  - *smurf (denial of service by flooding a network)"*

What is CVE?

- Common Vulnerabilities and Exposures (CVE)
- Created in 1999 by MITRE, Steve Christey Coley (@SushiDude) and David Mann
- https://cve.mitre.org/
- A *dictionary (not a database)* of common names (i.e., CVE Identifiers) for publicly known cybersecurity vulnerabilities
- Free for public download and use
- CVE ID Syntax: *CVE prefix + Year + Arbitrary Digits*
- Does NOT provide proof of concept (PoC) or exploit!

CVE-ID Syntax Change

Old Syntax
CVE-YYYY-NNNN
4 fixed digits, supports a maximum of 9,999 unique identifiers per year.

New Syntax
CVE-YYYY-NNNN...N
4-digit minimum and no maximum, provides for additional capacity each year when needed.

Fixed 4-Digit Examples
CVE-1999-0067
CVE-2005-4873
CVE-2012-0158

Arbitrary Digits Examples
CVE-2014-0001
CVE-2014-12345
CVE-2014-7654321

YYYY indicates year the ID is issued to a CVE Numbering Authority (CNA) or published.

Implementation date: January 1, 2014

Source: http://cve.mitre.org

Then There is CWE. What is CWE?

- Common Weakness Enumeration
- Also maintained by MITRE
- https://cwe.mitre.org/
- *"A formal list of software weakness types created to:*
  - *Serve as a common language for describing software security weaknesses in architecture, design, or code.*
  - *Serve as a standard measuring stick for software security tools targeting these weaknesses.*
  - *Provide a common baseline standard for weakness identification, mitigation, and prevention efforts."*
- Source: https://cwe.mitre.org/about/index.html

Tufts UNIVERSITY | SCHOOL OF ENGINEERING
Computer Science

What is CWE? Continued

- **Some Common Types of Software Weaknesses:**
  - Buffer Overflows, Format Strings, Etc.
  - Structure and Validity Problems
  - Common Special Element Manipulations
  - Channel and Path Errors
  - Handler Errors
  - User Interface Errors
  - Pathname Traversal and Equivalence Errors
  - Authentication Errors
  - Resource Management Errors
  - Insufficient Verification of Data
  - Code Evaluation and Injection
  - Randomness and Predictability

What's the Difference Between CVE and CWE?

- Arguably the best explanation via Daniel Miessler:
  - "CWE: has to do with the vulnerability—not the instance within a product or system"
  - "CVE: has to do with the specific instance within a product or system—not the underlying flaw."
  - Source: https://danielmiessler.com/blog/difference-cve-cwe/
- Example:
  - CVE-2015-2213 is a SQL injection vulnerability in WordPress
  - CWE-89: Improper Sanitization of Special Elements used in an SQL Command (is the weakness (or flaw) in the code of WordPress that caused CVE-2015-2213.)
  - Source: https://www.veracode.com/blog/2016/08/language-appsec

National Vulnerability Database

- https://nvd.nist.gov/home.cfm
- Maintained by NIST: National Institute of Standards and Technology
- Uses CVE
- Database; contains references to advisories, solutions, and tools
- Example, regarding CVE-2015-2213 (from previous slide): https://nvd.nist.gov/vuln/detail/CVE-2015-2213

Open Sourced Vulnerability Database (OSVDB)

- http://osvdb.org/
- People: attrition.org, H.D. Moore, Rain Forest Puppy, Chris Sullo
- DEAD on April 5th 2016, looking for someone to pick it back up
- Open source
- "OSVDB's goal is to provide accurate and unbiased information about security vulnerabilities in computerized equipment. The OSVDB blog discusses various topics related to vulnerabilities including disclosure, running a vulnerability database (VDB), and more." (https://blog.osvdb.org/about/)

The Exploit Database

- https://www.exploit-db.com/
- Maintained by Offensive Security
- A CVE compliant archive of exploits and vulnerable software
- "A repository for *exploits and proof-of-concepts rather than advisories*"
  - Source: https://www.exploit-db.com/about/
- Downloadable
  - Tool: `searchsploit` - command line search tool for Exploit-DB

Scanning for Vulnerabilities

- Tools:
  - Nikto
  - Nessus
  - OpenVAS
  - Metasploit
  - w3af
  - Many others

Tool: Nikto

- Written by Chris Sullo

- Open Source

- Web server scanner "designed to find various default and insecure files, configurations and programs on any type of web server"

- Documentation: https://cirt.net/nikto2-docs/

- Source code: https://github.com/sullo/nikto

- Example: `nikto --host <IP ADDRESS>`

Tool: Nikto (example continued)

```
$ nikto --host=192.168.1.66 --root=/mutillidae

- Nikto v2.1.5
---------------------------------------------------------------------
+ Target IP:          192.168.1.66
+ Target Hostname:    192.168.1.66
+ Target Port:        80
+ Target Path:        /mutillidae
+ Start Time:         2017-06-05 18:13:27 (GMT-4)
---------------------------------------------------------------------
+ Server: nginx/1.6.2
+ Cookie PHPSESSID created without the httponly flag
+ Cookie showhints created without the httponly flag
+ The anti-clickjacking X-Frame-Options header is not present.
+ Uncommon header 'logged-in-user' found, with contents:
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /mutillidae/robots.txt, fields: 0x586fe41c 0
xbe
+ "robots.txt" contains 8 entries which should be manually viewed.
+ /mutillidae/index.php?page=../../../../../../../../../etc/passwd: The PHP-Nuke Rocket add-in is
 vulnerable to file traversal, allowing an attacker to view any file on the host. (probably Rocket,
but could be any index.php)
+ OSVDB-3233: /mutillidae/phpinfo.php: Contains PHP configuration information
+ OSVDB-12184: /mutillidae/index.php?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potential
ly sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3093: /mutillidae/.htaccess: Contains authorization information
+ OSVDB-5292: /mutillidae/?_CONFIG[files][functions_page]=http://cirt.net/rfiinc.txt?: RFI from RSna
ke's list (http://ha.ckers.org/weird/rfi-locations.dat) or from http://osvdb.org/
+ OSVDB-5292: /mutillidae/?npage=-1&content_dir=http://cirt.net/rfiinc.txt?%00&cmd=ls: RFI from RSna
ke's list (http://ha.ckers.org/weird/rfi-locations.dat) or from http://osvdb.org/
+ OSVDB-5292: /mutillidae/?npage=1&content_dir=http://cirt.net/rfiinc.txt?%00&cmd=ls: RFI from RSnak
```

Tool: Nessus

- Commercial
- Was open source at one point; closed source in 2005
- "The world's most widely deployed vulnerability scanner"
- "There are 86731 plugins, covering 38201 unique CVE IDs and 25042 unique Bugtraq IDs." https://www.tenable.com/plugins/index.php?view=all
- https://www.tenable.com/products/nessus-vulnerability-scanner
- Nessus Home (free): https://www.tenable.com/products/nessus-home
- OpenVAS (Open Vulnerability Assessment System) is a free and open source fork of Nessus

Tool: Nessus (continued)

Tool: Metasploit

- https://www.metasploit.com/
- Source code: https://github.com/rapid7/metasploit-framework
- Written by H.D. Moore
- Acquired by Rapid7 in 2009
- Open-source platform for developing, testing, and using exploit code
- Currently has over 1600 exploits, 400 payloads

Tool: Metasploit (continued)

To Ponder

- Question: If you do a scan or a penetration test of a system and no vulnerabilities are reported, is that a good thing?

- Source of picture: Gary McGraw

Badness-ometer != security meter

DEEP TROUBLE

WHO KNOWS

badness-ometer

The Vulnerability Disclosure Debate

- **Vulnerability disclosure** – the practice of reporting security bugs or flaws in a computer software or hardware

- Debate has raged on for decades

- Options:
  - Full disclosure
  - No disclosure
  - Responsible disclosure

To Ponder

- Question 1: say you have found a known vulnerability on a system or software. How would you disclose the vulnerability and to whom? What could possibly go wrong?

- Question 2: say you have found an unknown vulnerability, a vulnerability not known by the public including companies (a.k.a., **zero day**), on a system or software. How would you disclose the vulnerability and to whom? What could possibly go wrong?