



Counterattack

Turning the tables on
exploitation attempts from tools
like Metasploit

whoami

- scriptjunkie
 - Security research
 - Metasploit contributor

whoami

- wrote this thing...

msfgui

hosts

host type via exploit via payload

127.0.0.1 meterpreter exploit/multi

Activities

Nov 24, 2010 4:59:41 PM msfgui sta

Nov 24, 2010 4:59:44 PM Session 1

Nov 24, 2010 5:07:00 PM msfgui log

Session logs

Session 1

127.0.0.1:4444

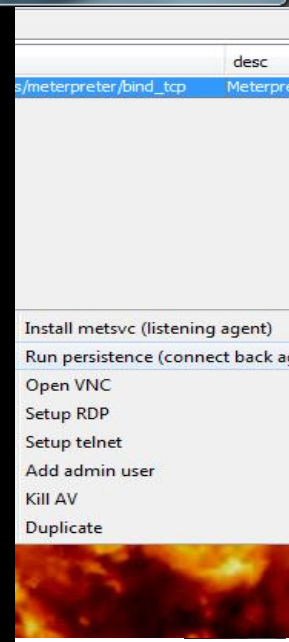
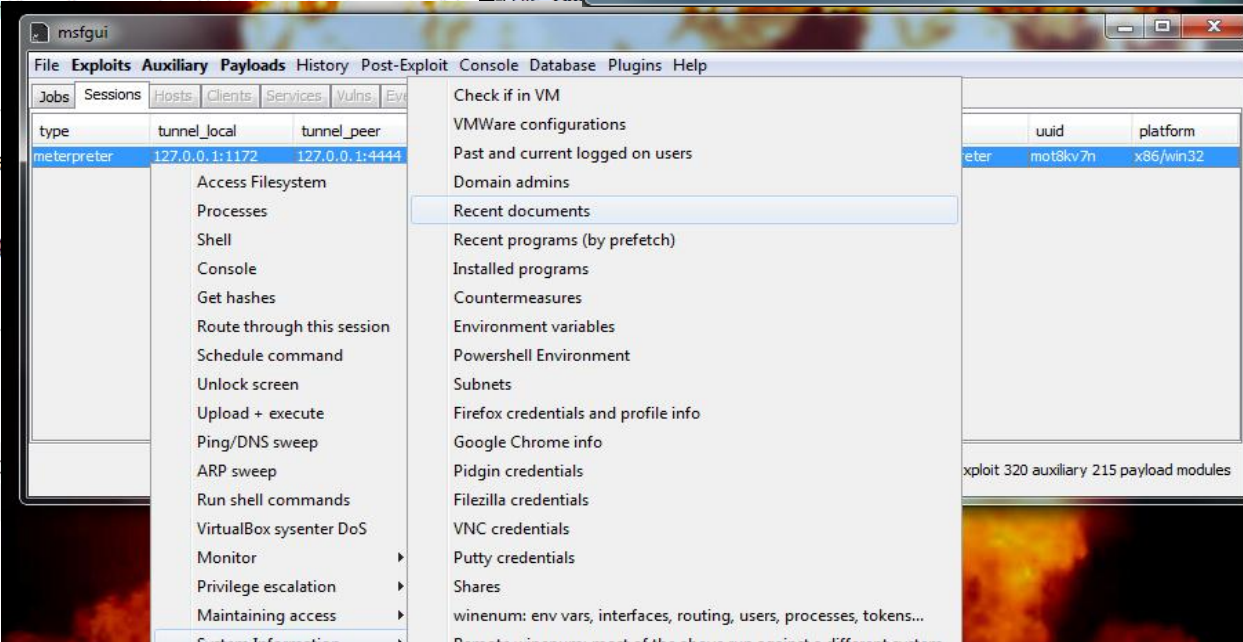
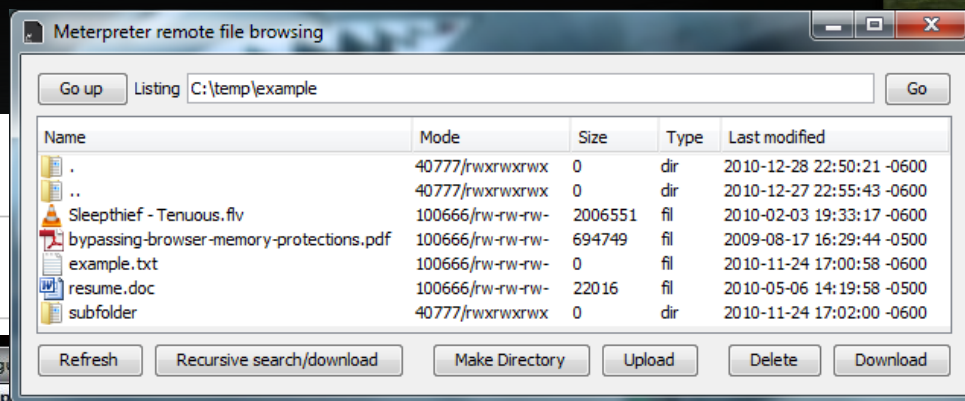
Nov 24, 2010 5:02:23 PM >>>cd "C:

Nov 24, 2010 5:02:23 PM >>>ls

Nov 24, 2010 5:02:23 PM

Listing: C:\

=====



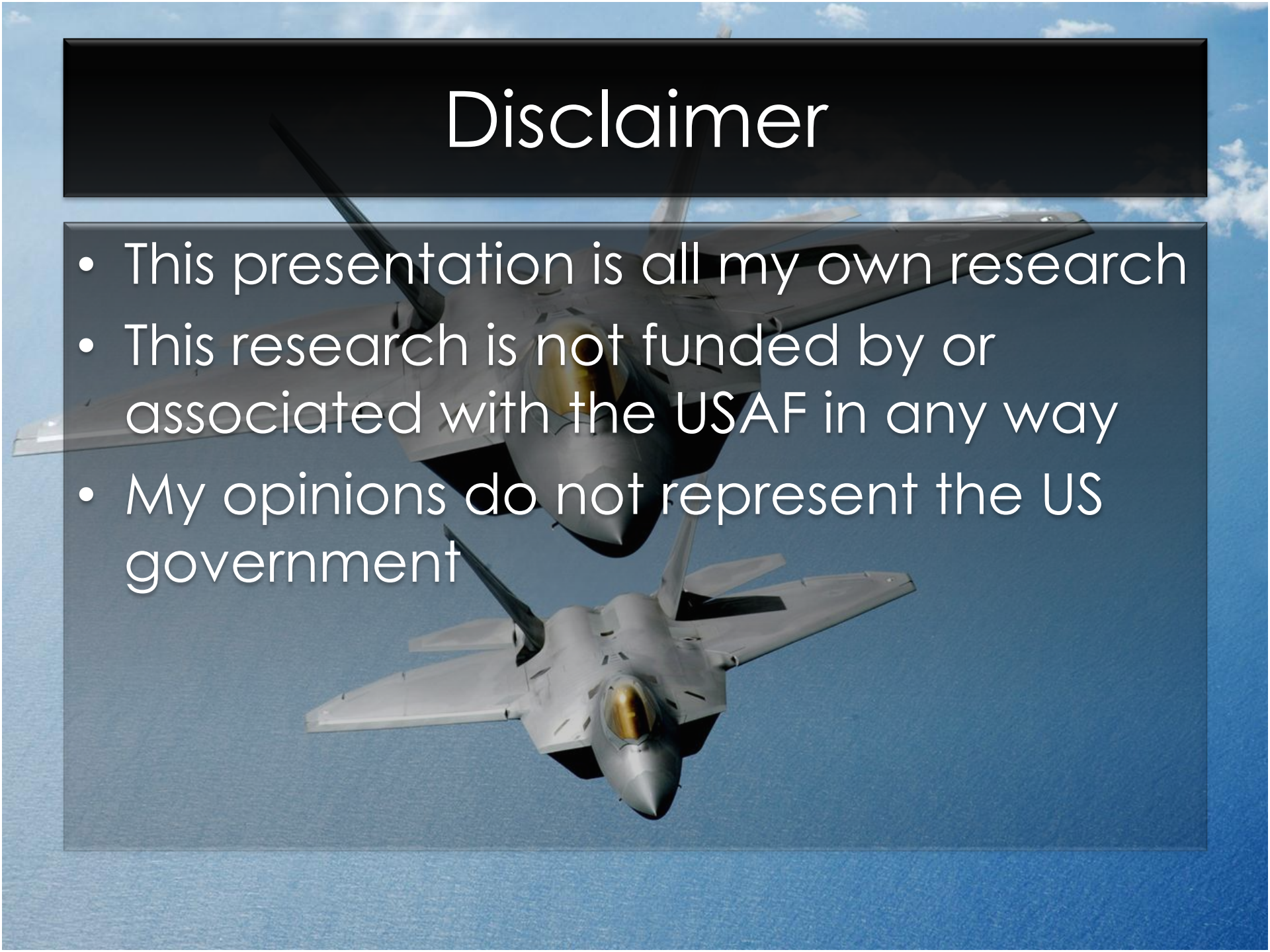
whoami

- I work here



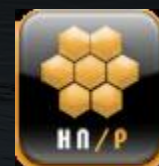
Disclaimer

- This presentation is all my own research
- This research is not funded by or associated with the USAF in any way
- My opinions do not represent the US government



Previous work

- Honeypots



Previous Work

- Backtrack vulnerabilities...
 - Rob DeGulielmo, “Con Kung-Fu” DC17

Exploit pack Exploits

- LuckySploit, UniquePack referrer XSS
 - Paul Royal, Purewire, August 2009
- Zeus
 - BK, xs-sniper.com Sept 2010



Ethics

- Some ideas:
 - Self-defense
 - Neutralizing
 - Unintended Consequences
 - Worms
- Left as an exercise for the student

Generic Counterattacks

- Worms
 - Get weaponized version of exploit
 - Neutralize attacking systems
 - Be careful!

Windows Counterattacks

- SMB is your friend
- Getting attackers to bite
 - May require IE
 - Vulnerable-looking web pages that only work on IE 6?
- SMB relay FTW!
- Or at least capture

Demo



Popular security tools

- Nmap
- Firesheep
- Nessus
- Cain & Abel
- Snort
- Wireshark
- Metasploit

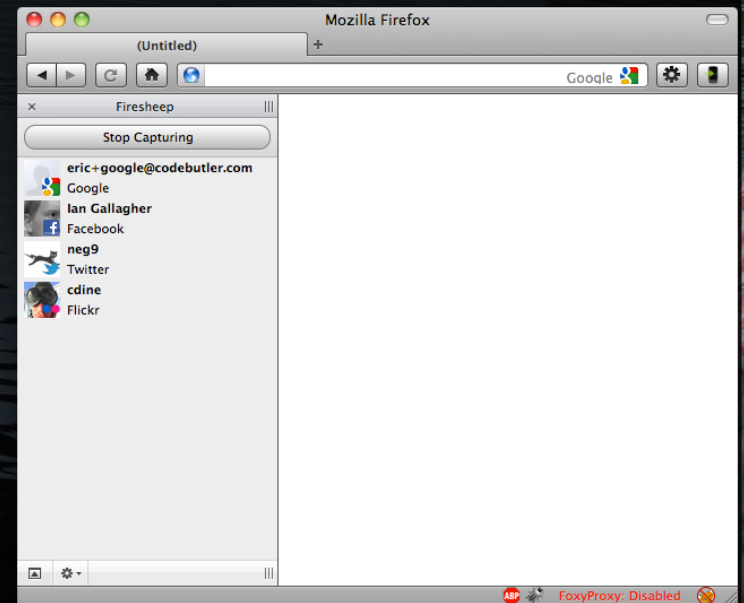
Nmap

- No RCE
- Can still mislead
- Open ports
- Tar pits
- DoS
- Demo



Firesheep

- And then there's blacksheep to detect
- And there's fireshepherd to DoS



Nessus

- CVE-2010-2989
 - nessusd_www_server.nbin in the Nessus Web Server plugin 1.2.4 for Nessus allows remote attackers to obtain sensitive information via a request to the /feed method.
- CVE-2010-2914
 - Cross-site scripting (XSS) vulnerability in nessusd_www_server.nbin in the Nessus Web Server plugin 1.2.4 for Nessus.
- ...





Password Recovery
Utility



Cain & Abel

- CVE-2005-0807
 - Multiple buffer overflows in Cain & Abel before 2.67 allow remote attackers to cause a denial of service (application crash) and possibly execute arbitrary code via (1) an IKE packet with a large ID field that is not properly handled by the PSK sniffer filter, (2) the HTTP sniffer filter, or the (3) POP3, (4) SMTP, (5) IMAP, (6) NNTP, or (7) TDS sniffer filters.
- CVE-2008-5405
 - Stack-based buffer overflow in the RDP protocol password decoder in Cain & Abel 4.9.23 and 4.9.24, and possibly earlier...

Snort

- CVE-2009-3641
 - Snort before 2.8.5.1, when the -v option is enabled, allows remote attackers to cause a denial of service (application crash) via a crafted IPv6 packet that uses the (1) TCP or (2) ICMP protocol.
- CVE-2008-1804
 - preprocessors/spp_frag3.c in Sourcefire Snort before 2.8.1 does not properly identify packet fragments that have dissimilar TTL values, which allows remote attackers to bypass detection rules by using a different TTL for each fragment.



Wireshark



- CVE-2010-4301
 - epan/dissectors/packet-zbee-zcl.c in the ZigBee ZCL dissector in Wireshark 1.4.0 through 1.4.1 allows remote attackers to cause a denial of service (infinite loop) via a crafted ZCL packet...
- CVE-2010-4300
 - Heap-based buffer overflow in the dissect_ldss_transfer function (epan/dissectors/packet-ldss.c) in the LDSS dissector in Wireshark 1.2.0 through 1.2.12 and 1.4.0 through 1.4.1 ...

Wireshark

- Vulnerabilities!
 - 100's of protocol dissectors
 - Non memory-safe language
 - Usually run as root on linux
 - Build a fuzzer!

Wireshark

- Or just look it up



Wireshark Bug Database – Bug List

Wireshark Home | New | Search | Find | Reports | Requests | Help | New Account | Log In | Forgot Password

Wed Dec 1 2010 17:12:04 PST

Status: NEW, ASSIGNED, REOPENED

559 bugs found.

ID	Sev	Pri	OS	Assignee	Status	Resolution	Summary
2579	Cri	Hig	Wind	wireshark-bugs	NEW		u3 version not working (winpcap missing error)
4205	Cri	Hig	Mac	wireshark-bugs	NEW		OSX build on 10.6.1 will not start or load
4569	Cri	Low	Ubun	wireshark-bugs	NEW		Trying to run dumpcap as "root" causes system to completely lockup (requires hard reset)
4645	Cri	Low	Wind	wireshark-bugs	NEW		Wireshark Crashing by pressing 2 Buttons
5112	Cri	Low	Wind	wireshark-bugs	NEW		wireshark will crash if a specific mib being added to SMI path and module but missing some dependent other mib
5284	Cri	Hig	Ubun	wireshark-bugs	NEW		new_packet_list: redissection + redraw crashes when multi-data-source packet is selected

Wireshark

- Stack traces at no extra charge!

Wireshark on Fedora 14/x86_64 crashes when loading a capture that I've captured on a XP/32 with 1.4.1

I can't provide the capture, but here's the stack. I may be able to get just the offending packet - if I knew what it was:

```
#0 0x000000384b449612 in _IO_vfprintf_internal (s=<value optimized out>,
format=<value optimized out>, ap=<value optimized out>) at vfprintf.c:1561
#1 0x000000384b4faf30 in __vsnprintf_chk (s=0x7fff76ad0abf "", maxlen=<value
optimized out>, flags=1, slen=<value optimized out>, format=
0x7f522d73454a "(%s=%s)", args=0x7fff76ad0b20) at vsnprintf_chk.c:65
#2 0x000000384dc4a573 in vsnprintf (format=<value optimized out>, args=<value
optimized out>) at /usr/include/bits/stdio2.h:78
#3 g_printf_string_upper_bound (format=<value optimized out>, args=<value
optimized out>) at gmessages.c:1109
#4 0x00007f522c7d2749 in ep_strdup_vprintf (fmt=0x7f522d73454a "(%s=%s)",
ap=<value optimized out>) at emem.c:883
#5 0x00007f522c7d281d in ep_strdup_printf (fmt=<value optimized out>) at
emem.c:899
#6 0x00007f522cf39e1c in dissect_ldap_T_equalityMatch (implicit_tag=<value
optimized out>, tvb=<value optimized out>, offset=37,
actx=<value optimized out>, tree=<value optimized out>, hf_index=<value
optimized out>) at ldap.cnf:536
#7 0x00007f522c8bf23b in dissect_ber_choice (actx=0x7fff76ad1450,
parent_tree=0x0, tvb=0x2a0ba40, offset=<value optimized out>, choice=
0x7f522de37640, hf_id=35463, ett_id=9811, branch_taken=0x0) at
packet-ber.c:3013
#8 0x00007f522cf3b222 in dissect_ldap_Filter (tvb=0x2a0ba40, offset=0,
actx=0x7fff76ad1450, tree=0x0, hf_index=35463,
implicit_tag=<value optimized out>) at ldap.cnf:686
#9 0x00007f522cf3b364 in dissect_ldap_T_and_item (implicit_tag=<value
```

Wireshark

- And fuzzers come for free!

From: bugzilla-daemon@xxxxxxxxxxxxx
Date: Sun, 28 Nov 2010 11:50:07 -0800 (PST)

https://bugs.wireshark.org/bugzilla/show_bug.cgi?id=5448

Summary: Buildbot crash output: fuzz-2010-11-28-11164.pcap
Product: Wireshark
Version: unspecified
Platform: x86-64
URL: <http://www.wireshark.org/download/automated/captures/fuzz-2010-11-28-11164.pcap>
OS/Version: Ubuntu
Status: NEW
Severity: Critical
Priority: High
Component: TShark
AssignedTo: wireshark-bugs@xxxxxxxxxxxxx
ReportedBy: buildbot-do-not-reply@xxxxxxxxxxxxx

Build Information:

--

Problems have been found with the following capture file:

<http://www.wireshark.org/download/automated/captures/fuzz-2010-11-28-11164.pcap>

stderr:

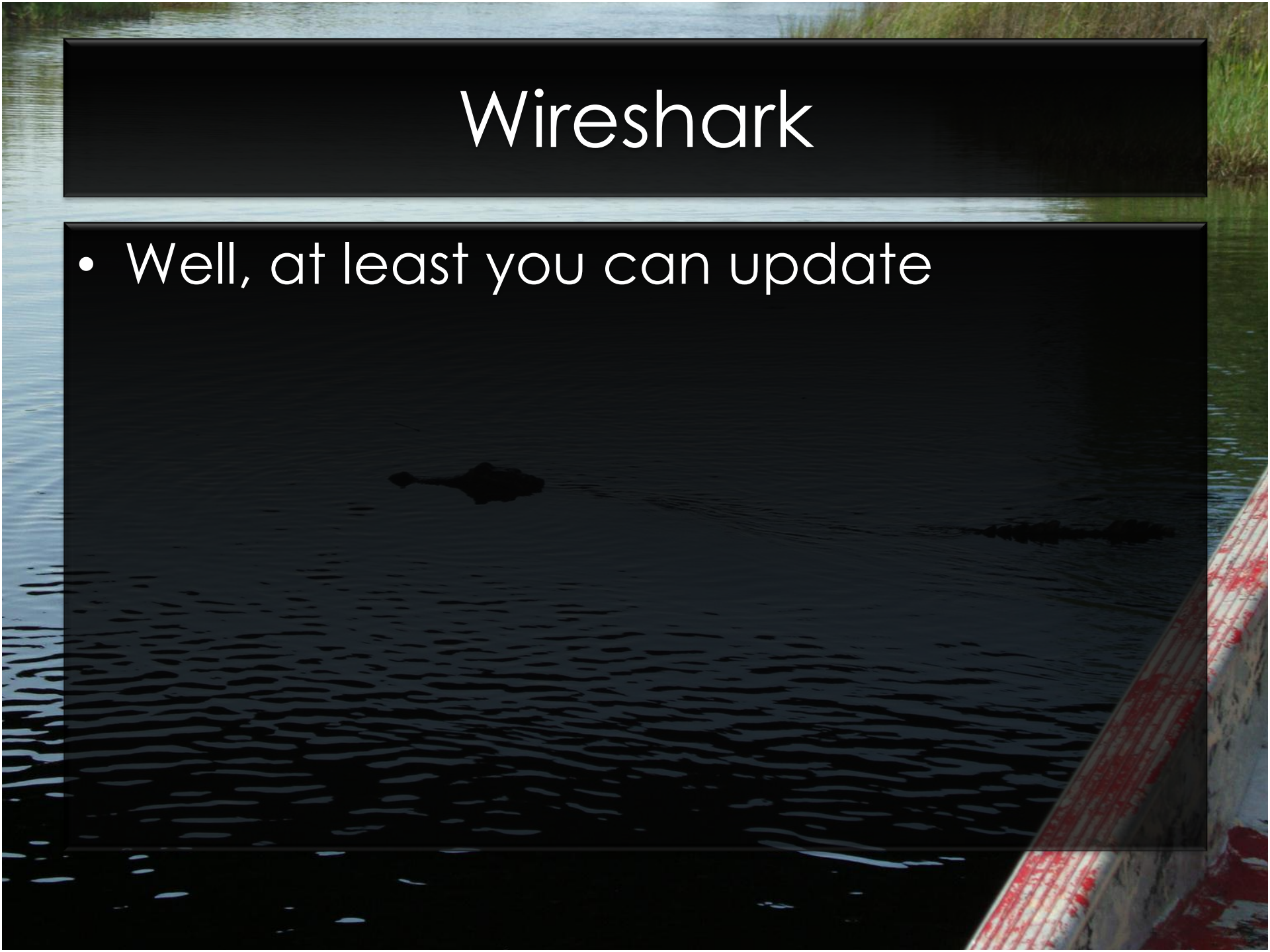
*** stack smashing detected ***: ./tshark terminated

===== Backtrace: =====

/lib/libc.so.6(__fortify_fail+0x37) [0x7f889652d217]
/lib/libc.so.6(__fortify_fail+0x0) [0x7f889652d1e0]
/home/wireshark/builders/trunk/ubuntu1004x64/install/lib/libwireshark.so.0(+0x11a9ad9) [0x7f8898725ad9]
/home/wireshark/builders/trunk/ubuntu1004x64/install/lib/libwireshark.so.0(+0x11ac010) [0x7f8898728010]
/home/wireshark/builders/trunk/ubuntu1004x64/install/lib/libwireshark.so.0(+0xf01940) [0x7f889847d940]
/home/wireshark/builders/trunk/ubuntu1004x64/install/lib/libwireshark.so.0(+0xf0206d) [0x7f889847e06d]
/home/wireshark/builders/trunk/ubuntu1004x64/install/lib/libwireshark.so.0(dissector_try_port_new+0x61)
/home/wireshark/builders/trunk/ubuntu1004x64/install/lib/libwireshark.so.0(+0x11e5e86) [0x7f8898761e86]

Wireshark

- Well, at least you can update



Wireshark

- Unless you can't

The screenshot shows the Wireshark download page and the Ubuntu Software Center. The top part is the 'Download Wireshark' page, which includes a 'Get Wireshark' section. It states that the current stable release is 1.4.2, superseding all previous releases, including all releases of [Ethereal](#). It also mentions the latest development release (1.4.0rc2) and documentation. Below this, there are two buttons: 'Stable Release (1.4.2)' and 'Old Stable Release (1.2.13)'. The bottom part of the screenshot shows the Ubuntu Software Center interface. It has a top bar with 'Manager' and 'Help' buttons. Below that is a toolbar with 'Apply' and 'Properties' buttons. The main area is a table listing installed and available packages.

Download Wireshark

Get Wireshark

The current stable release of Wireshark is 1.4.2. It supersedes all previous releases, including [all releases of Ethereal](#). You can also download the latest development release (1.4.0rc2) and documentation.

► Stable Release (1.4.2)

▼ Old Stable Release (1.2.13)

Manager

Help

es Apply Properties

▼	Package	Installed Version	Latest Version	Description
■	wireshark	1.2.11-2	1.2.11-2	network traffic
■	wireshark-common	1.2.11-2	1.2.11-2	network traffic

Metasploit

METASPLOIT



Finding vulnerabilities

- or - Why not fuzz?

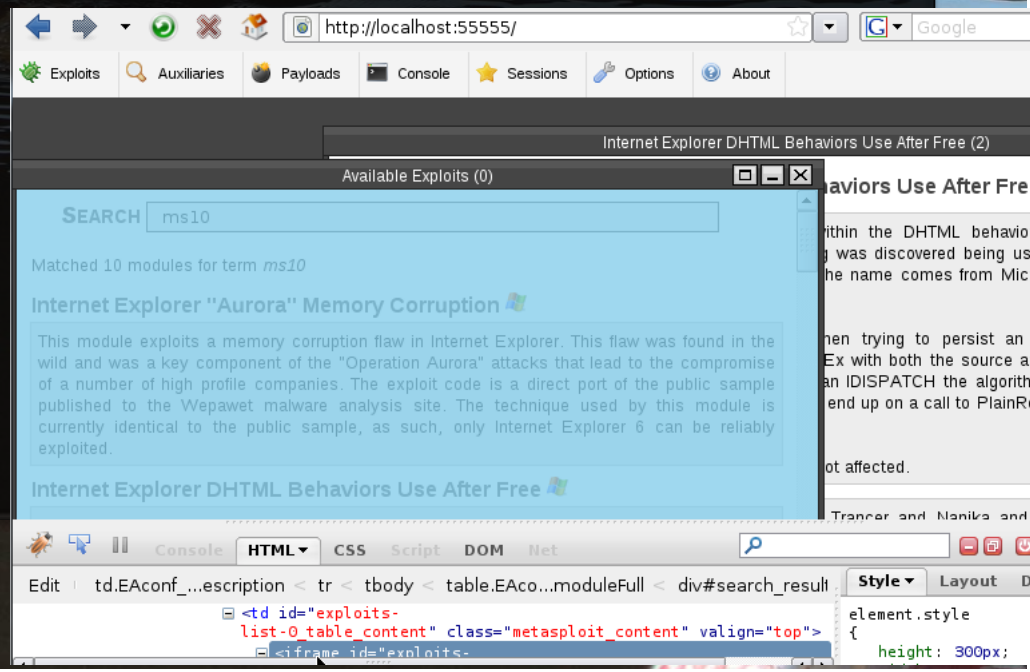
- Memory corruption
 - Openssl?
 - Ruby
- Logic errors

Web UI

- Things get more interesting
- Classic webapp attacks up for grabs
- Control of msfweb = control of metasploit
- Control of metasploit = control of system

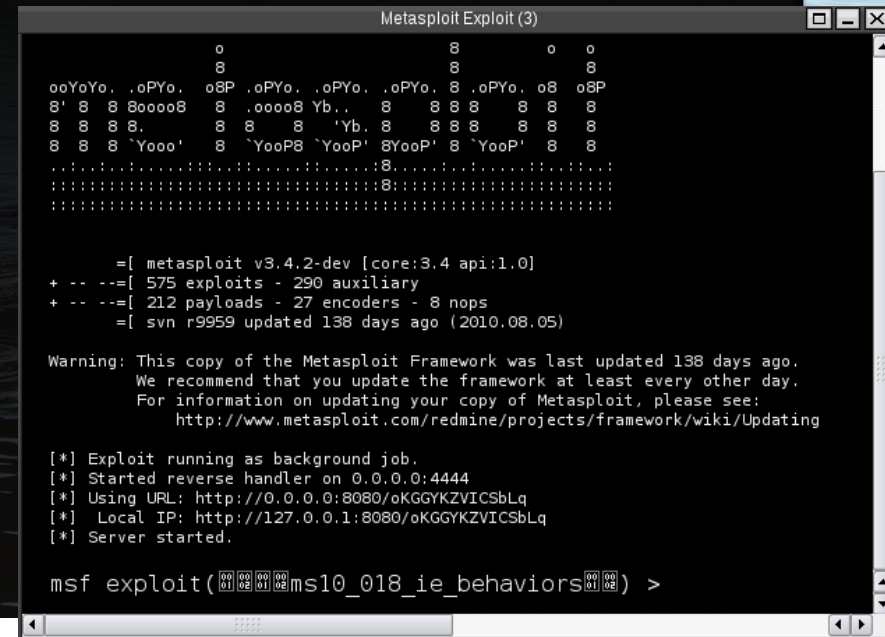
Web UI Structure

- Frame based module launching
- Available Exploits -> Select Target -> Select Payload -> Options -> Launch
- Server is stateless
- Until launch
- /exploits/config post with options



Web UI

- New console creation from module
- /console/index/0
- /console/index/1 ...
- Request to /console manually creates
- Polls for output



```
Metasploit Exploit (3)

      o               8      o      o
      8               8      8      8
ooYoYo. .oPYo. .o8P .oPYo. .oPYo. .oPYo. .oPYo. o8 o8P
8' 8 8 8oooo8 8 .oooo8 Yb.. 8 8 8 8 8 8 8
8 8 8 8.      8 8 8 8 'Yb. 8 8 8 8 8 8 8
8 8 8 `Yooo' 8 `YooP8 `YooP' 8YooP' 8 `YooP' 8 8
.....8.....
:::8:::
:::8:::

      =[ metasploit v3.4.2-dev [core:3.4 api:1.0]
+ -- ==[ 575 exploits - 290 auxiliary
+ -- ==[ 212 payloads - 27 encoders - 8 nops
      =[ svn r9959 updated 138 days ago (2010.08.05)

Warning: This copy of the Metasploit Framework was last updated 138 days ago.
We recommend that you update the framework at least every other day.
For information on updating your copy of Metasploit, please see:
      http://www.metasploit.com/redmine/projects/framework/wiki/Updating

[*] Exploit running as background job.
[*] Started reverse handler on 0.0.0.0:4444
[*] Using URL: http://0.0.0.0:8080/oKGGYKZVICSbLq
[*] Local IP: http://127.0.0.1:8080/oKGGYKZVICSbLq
[*] Server started.

msf exploit(000001ms10_018_ie_behaviors0000) >
```

EXITFUNC	Required	process
Exit technique: seh, thread, process (type: raw)		
LHOST	Required	0.0.0.0
The listen address (type: address)		
LPORT	Required	4444
The listen port (type: port)		
Launch Exploit		
ADVANCED OPTIONS		
ContextInformationFile		

Web UI Console

- Disabled commands
 - irb
 - System commands
- Reliability issues
 - Commands occasionally fail

Web UI Features

- Payload generation
- Frame sequence/option processing like exploits

Windows Meterpreter (Reflective Injection), Reverse TCP Stager (2)

External references:

- <http://www.harmonysecurity.com/ReflectiveDllInjection.html>

Size: 290
Architecture: x86
Operating system: Windows

OPTIONS

LHOST Required
The listen address (type: address)

EXITFUNC Required
Exit technique: seh, thread, process (type: raw)

LPORT Required
The listen port (type: port)

Max Size:

Restricted Characters (format: 0x00 0x01):

Selected Encoder:

Format:

C
Ruby
Perl
javascript
java

First Vulnerability

- Reflected XSS in payload generation
- Your encoded payload is displayed in a textarea
- Stars to align:
 - Payload must reflect arbitrary content (can't use normal shell/meterpreter payloads)
 - Encoder must generate predictable output (can't use most encoders, like shikata ga nai)
 - Format must preserve output (all listed formats only display hex of encoded payload)

XSS

- Payload cmd/unix/generic reflects arbitrary content
- Encoder generic/none leaves payload intact
- Payload format still works as a filter
 - Ruby, Java, Javascript, C arrays

XSS

- Unless you use an unlisted format
 - raw fmt + generic/none encoder + generic CMD payload = XSS

<http://localhost:55555/payloads/view?badchars=&commit=Generate&encoder=generic%2Fnone&refname=cmd%3Aunix%3Ageneric&step=1&format=raw>

- Inserted into

`<textarea> ... </textarea>`

- XSS!

`</textarea><script>alert(1)</script>`

Vulnerability Impact

- No ; or = or , allowed
- Eval, String.fromCharCode first stage
- XSS console control
- Getting RCE
 - Command injection
 - Metasploit

Vulnerability Impact

- Getting RCE
 - Key command – loadpath
 - Downloading a file
 - Servers
 - Meterpreter

Meterpreter

- Connection process
 - Stager connections
 - SSL
 - Initial request
 - Plugins
 - Command flow

Meterpreter

- Packet structure
 - TLV's

Meterpreter

- Packet structure
 - TLV's

Length	Type	Value
+-----+	+-----+	+-----+ ... +-----+

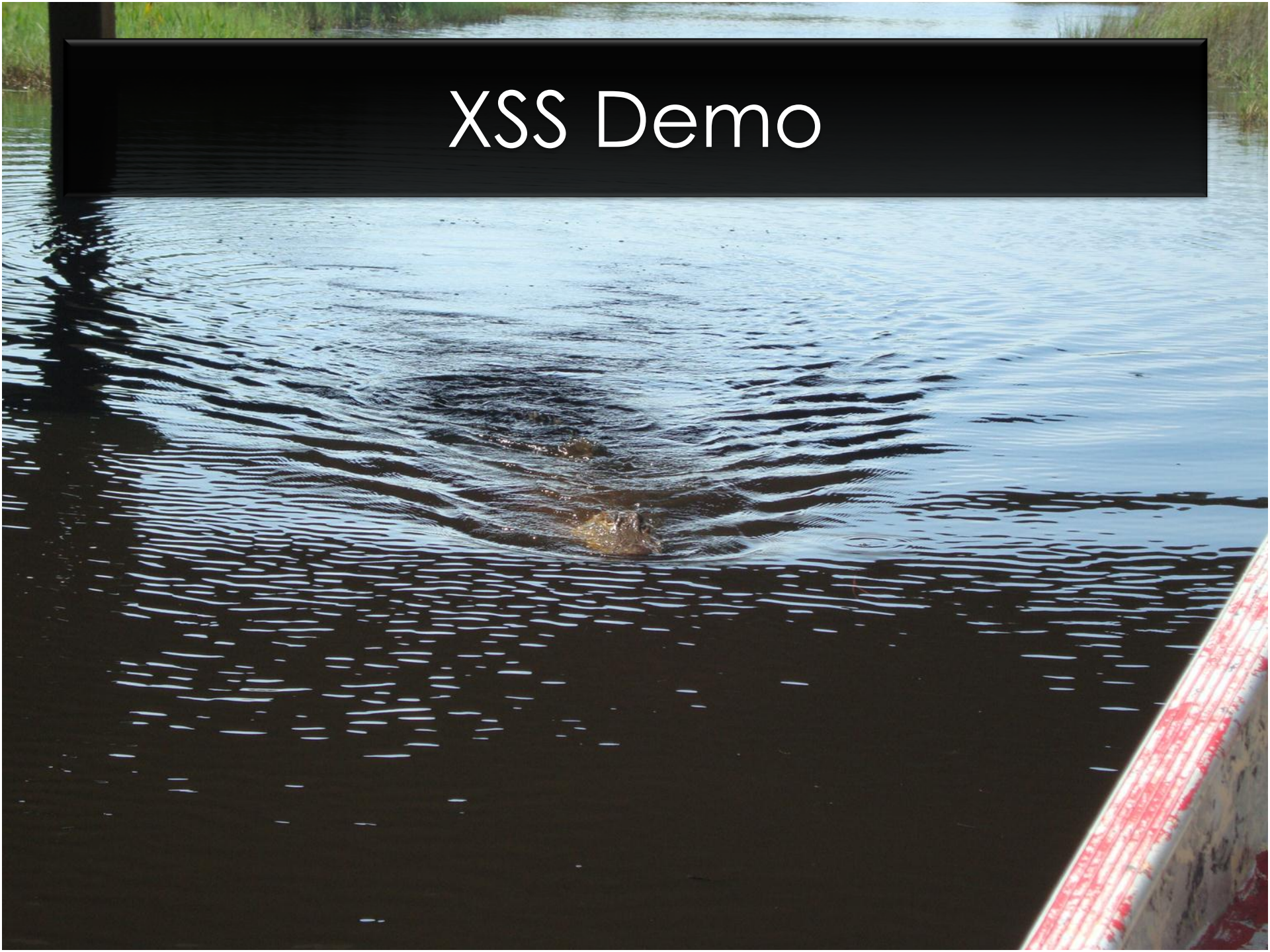
Meterpreter debugger

- View each TLV packet sent or received decoded
- Get all the information needed to emulate meterpreter calls

Exploit release

- XSS
 - Creates console
 - Launches meterpreter payload handler
 - Downloads ruby payload file
 - Loads ruby code
- Fake meterpreter to host shellcode
- Targets for all your favorite platforms

XSS Demo



Command Injection

- auxiliary/scanner/http/sqlmap
 - Is a special module
 - Options compose command line

```
63 # Test a single host
64 def run_host(ip)
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84 cmd = sqlmap + ' -u \'' + sqlmap_url + '\''
85 cmd += ' --method ' + method
86 cmd += ' ' + datastore['OPTS']
87
88
89
90
91
92
93
94
95
96
97 IO.popen( cmd ) do |io|
```

```
63 # Test a single host
64 def run_host(ip)
65
66 sqlmap = datastore['SQLMAP_PATH']
67
68 if not sqlmap
69   print_error("The sqlmap script could not be found")
70   return
71 end
72
73 data = datastore['DATA']
74 method = datastore['METHOD'].upcase
75
76 sqlmap_url = (datastore['SSL'] ? "https" : "http")
77 sqlmap_url += "://" + wmap_target_host + ":" + wmap_target_port
78 sqlmap_url += "/" + datastore['PATH']
79
80 if method == "GET"
81   sqlmap_url += '?' + datastore['QUERY']
82 end
83
84 cmd = sqlmap + ' -u \'' + sqlmap_url + '\''
85 cmd += ' --method ' + method
86 cmd += ' ' + datastore['OPTS']
87
88 if not data.empty?
89   cmd += ' --data \'' + data + '\''
90 end
91
92 if datastore['BATCH'] == true
93   cmd += ' --batch'
94 end
95
96 print_status("exec: #{cmd}")
97 IO.popen( cmd ) do |io|
```

Command Injection

- Also have
 - auxiliary/fuzzers/wifi/fuzz_beacon.rb
 - auxiliary/fuzzers/wifi/fuzz_proberesp.rb

```
40 1.upto(3) do |i|  
41   x = `ping -c 1 -n #{datastore['PING_HOST']}`  
42   return true if x =~ /1 received/
```

CSRF Vulnerability

- Input validation?
- CSRF
- Single-shot
- Generating a console
 - Finding a console
 - Reliable RCE metepreter-style difficult

CSRF Demo



Motivation

- I'm a Metasploit developer
- These were never patched
- Why release? Why not just fix the problems?
 - Maintainability
 - Disclosures

Meterpreter Vulnerability

- Meterpreter download process:
meterpreter> download foo
- In lib/rex/post/meterpreter/ui/console/
command_dispatcher/stdapi/fs.rb

```
stat = client.fs.file.stat(src)

if (stat.directory?)
  client.fs.dir.download(dest, src, recursive, true) { |step, src, dst|
    print_status("#{step.ljust(11)}: #{src} -> #{dst}")
  }
elsif (stat.file?)
  client.fs.file.download(dest, src) { |step, src, dst|
    print_status("#{step.ljust(11)}: #{src} -> #{dst}")
  }
end
```

Meterpreter Vulnerability

- File is saved as its basename
- In lib/rex/post/meterpreter/extensions/stdapi/fs/file.rb

```
69 def File.basename(*a)
70   path = a[0]
71   sep  = "\\\" + File::SEPARATOR
72
73   # I suck at regex.
74   path =~ /(.*?)#{sep}(.*?)$/
75
76   return $2 || path
77 end
```

Meterpreter Vulnerability

- Filtering out directory traversal

```
irb(main):001:0> path = "../../../traverse"  
=> "../../../traverse"  
irb(main):002:0> sep = "\\\" + File::SEPARATOR  
=> "\\\"  
irb(main):003:0> path =~ /(.*){sep}(.*)$/  
=> 0  
irb(main):004:0> $2  
=> "traverse"
```

Meterpreter Vulnerability

- ~~Filtering out~~ directory traversal
- File::SEPARATOR == "/" even on Windows!

```
irb(main):005:0> path = "../..\\..\\..\\traverse"  
=> "../..\\..\\..\\traverse"  
irb(main):006:0> path =~ /(.*){sep}(.*)$/  
=> 0  
irb(main):007:0> $2  
=> "..\\..\\..\\traverse"
```

Meterpreter Vulnerability

- But nobody's going to type
"download ../../../../../../evil"
- But they might type
"download juicydirname"
- Directories will take children with them

Meterpreter Traversal Demo



TFTP server

- Getting basename for file upload:
 - `tr[:file][:name].split(File::SEPARATOR)[-1]`

```
irb(main):002:0> path="../../../boot.ini"
=> "../../../boot.ini"
irb(main):003:0> path.split(File::SEPARATOR)[-1]
=> "boot.ini"

irb(main):004:0> path="../../../boot.ini"
=> "../../../boot.ini"
irb(main):005:0> path.split(File::SEPARATOR)[-1]
=> "boot.ini"
```

TFTP Traversal Demo



FTP server

- Directory traversal filtering

```
path = ::File.join(datastore['FTPROOT'], arg.gsub("../", '').gsub("../\\", ''))
```

```
irb(main):001:0> path="../../../etc/passwd"  
=> "../../../etc/passwd"  
irb(main):002:0> path.gsub("../", '').gsub("../\\", '')  
=> "etc/passwd"
```

FTP server

- Directory traversal filtering

```
irb(main):003:0> path="...//...//../etc/passwd"  
=> ".....//.....//../etc/passwd"  
irb(main):004:0> path.gsub("../", '').gsub("../\\", '')  
=> ".../.../etc/passwd"
```

Irony

- titanftp_xcrc_traversal.rb
- FTP traversal exploit with CRC brute force
- Byte-by-byte decode via XCRC command

FTP Traversal Demo



Scripts

- Often use client system name for log files

```
info = @client.sys.config.sysinfo
# Create Filename info to be appended to downloaded files
filenameinfo = "_" + ::Time.now.strftime("%Y%m%d.%M%S")

# Create a directory for the logs
logs = ::File.join(Msf::Config.log_directory, 'scripts',
'arp_scanner', info['Computer'] + filenameinfo)
# Create the log directory
::FileUtils.mkdir_p(logs)

#log file name
dest = logs + "/" + info['Computer'] + filenameinfo + ".txt"

print_status("Saving found IP's to #{dest}")
file_local_write(dest, found_ip)
```

Client system name

- Straight from not-to-be-trusted network data

```
request = Packet.create_request('stdapi_sys_config_sysinfo')
response = client.send_request(request)

{
    'Computer'      => response.get_tlv_value(TLV_TYPE_COMPUTER_NAME),
    'OS'            => response.get_tlv_value(TLV_TYPE_OS_NAME),
    'Architecture' => response.get_tlv_value(TLV_TYPE_ARCHITECTURE),
    'System Language' => response.get_tlv_value(TLV_TYPE_LANG_SYSTEM),
}
```

Scripts

- arp_scanner, domain_list_gen, dumplinks, enum_chrome, enum_firefox, event_manager, get_filezilla_creds, get_pidgin_creds, packetrecorder, persistence, search_dwld, winenum

domain_list_gen

- Counterattack can save file in arbitrary directory relative to home dir
- Starting with arbitrary contents

```
30 host = @client.sys.config.sysinfo['Computer']
31 current_user = client.sys.config.getuid.scan(/S*\((.*)/)
32 domain = @client.fs.file.expand_path("%USERDOMAIN%")
33 # Create Filename info to be appended to downloaded files
34 filenameinfo = "_" + ::Time.now.strftime("%Y%m%d.%M%S")
35 platform = client.platform.scan(/(win32|win64|php)/)
36 unsupported if not platform
37 # Create a directory for the logs
38 logs = ::File.join(Msf::Config.log_directory, 'scripts', 'domain_admins')
39 # Create the log directory
40 ::FileUtils.mkdir_p(logs)
41 #logfile name
42 dest = logs + "/" + host + filenameinfo + ".txt"
43 print status("found users will be saved to #{dest}")

73   file_local_write(dest, "#{domain}\\#{u}")
```

Lame DoS attacks

- Exploit handlers without ExitOnSession
- Meterpreter memory exhaustion
- Disk exhaustion: never-ending download

Writing Payloads

- Cross-platform RCE
 - Ruby is your friend
 - All msf libraries available for use
 - Can embed platform-specific or java payloads

Payloads

- New thread spinoff
- Multithreaded bind shell with error recovery
- Reverse shell with error handling

Wireshark Payloads

- Hard to do cross-platform
- Hard to do exploits cross-platform too
- Memory layouts, heap structures, system calls...

Persistence

- `~/.msf3/modules/exploits/`
 - Loaded on metasploit start, writeable by current user
 - Or payloads, auxiliary, encoders, nops
 - Ruby!
- `~/.msf3/msfconsole.rc`
 - Quasi-undocumented autorun resource file
 - Embeds ruby

Persistence

- Add something to main msf3 folder
 - /opt/metasploit3/msf3
 - C:\framework\msf3
- Relocate tree!
 - svn switch

Defenses



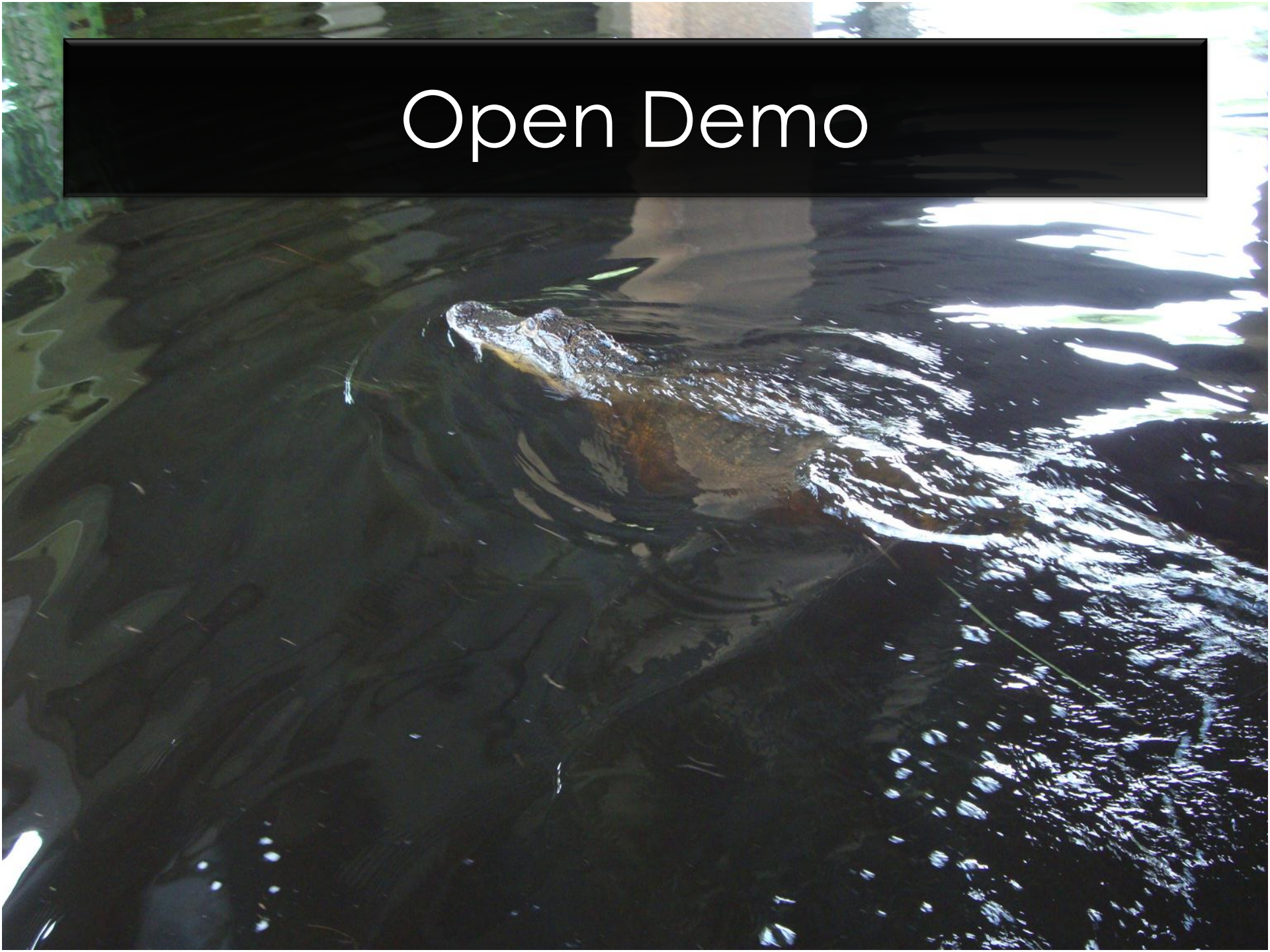
Defenses

- Developers/script writers
 - Don't trust input from the network
 - Don't trust client-side validation
 - Just because it looks like you control them doesn't mean it's true
- Users
 - Update!
- Limit privileges if possible
 - HTTP, SMB, DHCP, FTP, DNS, TFTP servers in Metasploit may require root
 - Most Nmap scans require root

Defenses

- Virtualization
 - Because VMs work
 - Saves privilege issues
 - Probably doesn't work with lorcon modules & raw wireless exploits
- OS choice

Open Demo



Wrap up

- Summary
- Lessons learned
- Products not shown here

Questions



A pixelated logo for 'METASPLOIT CYBERWAR' in blue and white on a black background. The text is arranged in two lines: 'METASPLOIT' on top and 'CYBERWAR' below it. A mouse cursor is visible over the 'E' in 'METASPLOIT'.

```
= [ metasploit v3.3.4-cyberwarfare [core:3.3 api:1.0]
+ -- -- [ 539 exploits - 260 auxiliary
+ -- -- [ 265 payloads - 23 encoders - 8 nops
      = [ svn r8974 updated today (2010.04.01)

msf > 
```