REDSPIN
INFORMATION SECURITY ASSESSMENTS

Beginner's Guide to the nmap Scripting Engine

David Shaw (dshaw@redspin.com)

REDSPIN
INFORMATION SECURITY ASSESSMENTS

First Things First

- These slides (and all code used) are available online at:

  http://github.com/davidshaw/toorcon2010

REDSPIN
INFORMATION SECURITY ASSESSMENTS

Who is this?

What is the NSE?

- Versatile lua framework for nmap

What is the NSE?

- Versatile lua framework for nmap

- Allows users to script scans natively

What is the NSE?

- Versatile lua framework for nmap

- Allows users to script scans natively

- Great at picking low-hanging fruit

Why lua?

Why lua?

- Fyodor likes it (isn't that enough?)

Why lua?

- Fyodor likes it (isn't that enough?)

- I prefer Ruby (Metasploit, anyone?), so I looked up some benchmarks

Benchmarks

Program Source Code	CPU secs	Elapsed secs	Memory KB	Code B	≈ CPU Load
pidigits					
Lua	2.80	**2.80**	1,648	414	0% 0% 0% 100%
Ruby 1.9	41.56	41.58	12,656	518	0% 0% 0% 100%
mandelbrot					
Lua	760.73	**760.69**	1,020	353	0% 0% 0% 100%
Ruby 1.9	5,852.08	5,849.73	2,848	313	0% 1% 0% 100%

Credit: http://shootout.alioth.debian.org/

REDSPIN
INFORMATION SECURITY ASSESSMENTS

What's already out there?

- There are **a lot** of awesome of scripts in every nmap install

- http://nmap.org/nsedoc/

What's already out there?

```
Starting Nmap 5.35DC1
  ( http://nmap.org ) at 2010-10-18
  15:02 PDT
NSE: Loaded 131 scripts for scanning.
```

REDSPIN
INFORMATION SECURITY ASSESSMENTS

What's already out there?

```
nmap <target> -sC all
```

```
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 21:11
Completed NSE at 21:11, 1.78s elapsed
Nmap scan report for www.microsoft.com (64.4.31.252)
Host is up (0.092s latency).
rDNS record for 64.4.31.252: wwwbay3vip.microsoft.com
Scanned at 2010-10-19 21:11:55 PDT for 2s
PORT    STATE SERVICE
443/tcp open  https
| sslv2: server still supports SSLv2
|        SSL2_DES_192_EDE3_CBC_WITH_MD5
|        SSL2_RC2_CBC_128_CBC_WITH_MD5
|        SSL2_RC4_128_WITH_MD5
|        SSL2_RC4_64_WITH_MD5
|        SSL2_DES_64_CBC_WITH_MD5
|_       SSL2_RC4_128_EXPORT40_WITH_MD5
```

Moving on...

- Great documentation at
  http://nmap.org/book/nse.html

REDSPIN
INFORMATION SECURITY ASSESSMENTS

Moving on...

- Great documentation at
  http://nmap.org/book/nse.html

- NSE's true power: anything you want

A Common Problem

- JMX Consoles

A Common Problem

- JMX Consoles

- Fairly common

REDSPIN
INFORMATION SECURITY ASSESSMENTS

A Common Problem

- JMX Consoles

- Fairly common

- Often run on non-standard ports, such as 8080

REDSPIN
INFORMATION SECURITY ASSESSMENTS

Internals of an NSE

- NSE's are simple (even if you don't code)

REDSPIN
INFORMATION SECURITY ASSESSMENTS

Internals of an NSE

- NSE's are simple (even if you don't code)

- Let's create one, line by line, from scratch

REDSPIN
INFORMATION SECURITY ASSESSMENTS

jmx_detect.nse

```
description = [[
This is an nmap script to search for
  accessible JMX web consoles.
]]
```

jmx_detect.nse

```
author = "David Shaw" -- hello, Toorcon!
```

jmx_detect.nse

```
author = "David Shaw" -- hello, Toorcon!

license = "see http://nmap.org/book/man-
  legal.htm"
```

jmx_detect.nse

```
author = "David Shaw" -- hello, Toorcon!

license = "see http://nmap.org/book/man-
  legal.htm"

categories = {"default", "discovery",
  "safe"}
```

- We want to trigger on certain ports, and JMX consoles are served over HTTP

```
require “shortport”
require “http”
```

portrule

- “portrule” lets us tell nmap when to trigger our script

- “shortport” further simplifies this process

portrule

- “portrule” lets us tell nmap when to trigger our script

- “shortport” further simplifies this process

```
portrule = shortport.port_or_service(
  {80, 443, 8080}, {“http”, “https”}
)
```

action

- The “action” function runs when portrule is matched

```
action = function(host, port)
 -- do stuff in here
end
```

action

```
action = function(host, port)
 -- we only care about the HTTP status (quick demo!)
 local stat = http.get(host, port, '/jmx-console/').status
end
```

action

```
action = function(host, port)
 -- we only care about the HTTP status (quick demo!)
 local stat = http.get(host, port, '/jmx-console/').status

 -- HTTP 200 (OK) means we probably found a JMX console!
 if stat == 200 then
   return "[+] Found possible JMX Console!"
 end
end
```

Bringing it all together

```
require 'http'
require 'shortport'
portrule = shortport.port_or_service({80, 443, 8080},
  {“http”, “https”})
action = function(host, port)
 local stat = http.get(host, port, '/jmx-console/').status
 if stat == 200 then
    return “[+] Found possible JMX Console!”
 end
end
```

REDSPIN
INFORMATION SECURITY ASSESSMENTS

Execution

```
code@dev:~/projects$ nmap 173.203.27.184 -p80 -PN --script jmx_detect.nse

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2010-10-20 21:56 PDT
Nmap scan report for 173.203.27.184
Host is up (0.037s latency).
PORT   STATE SERVICE
80/tcp open  http
|_jmx_detect: [+] Found possible JMX Console!

Nmap done: 1 IP address (1 host up) scanned in 16.68 seconds
```

REDSPIN
INFORMATION SECURITY ASSESSMENTS

We Did It!

Thank you to:

Fyodor & the nmap team

My incredible coworkers, past and present

(Mark, Joel, DB, Paul, Jason, Nate: that means you!)