Scanning

Goals

Useful Tools

The Basics

NMAP

Scanning
Suppose you’re an attacker
You want to attack a site
How do you proceed?
Goals

- Find an interesting (or vulnerable) machine
- Find a vulnerable service
- Attack...
Useful Tools

- Ping
- Arp
- Dig
- Nmap
- rpcinfo; showmount
- Tcpdump
- Others, for special purposes
Scanning

The Basics

Getting Started
What are the Hosts?
What Happened?
Enumerating Hosts
Other Information in the DNS
What Hosts Really Exist?
How About a Broadcast ping?
Off-LAN Broadcasts
ARP
NMAP

The Basics
Getting Started

- What’s the first thing we know about the target?
- The domain name!
- Your probably know at least one host, too: www.domainname
- There’s more in the DNS
What are the Hosts?

- Most hosts have DNS entries — can we list them?
- First try — do “zone transfer”
- Use `dig ns cs.columbia.edu to learn the name servers`
- Pick one, then

```bash
$ dig axfr cs.columbia.edu @dns2.itd.umich.edu

; <<>> DiG 9.3.2 <<>> axfr cs.columbia.edu @dns2.itd.umich.edu
; (1 server found)
;; global options: printcmd
; Transfer failed.
```

- But a different name server worked...
It’s possible to configure a name server to reject unauthorized zone transfer requests.

But most sites have multiple name servers; frequently, some are under different management (including 2 of 4 cs.columbia.edu name servers).

Not everyone has the same policy...
Enumerating Hosts

- Learn the IP address of one host: www.cs.columbia.edu is 128.59.23.100
- Use `dig -x` on other IP addresses in the range:

```bash
for i in `seq 1 254`
do
dig -x 128.59.23.$i
done
```

- Some sites give useless answers; 135.207.23.32 is H-135-207-23-32.research.att.com
- Another caveat: watch out for smaller or larger nets
Other Information in the DNS

- **HINFO:**

  ```
  $ dig hinfo play.cs.columbia.edu.
  play.cs.columbia.edu. 3600 IN
  ```

- **More:** see WKS records, TXT records, NAPTR records, etc.

  ```
  $ dig wks cs.columbia.edu
  cs.columbia.edu. 3600 IN WKS
  128.59.16.20 6 13 17 21 23 25 37 42 53 79
  111 119 67 69 161 162
  ```

- **Of course, those might be wrong...**
What Hosts Really Exist?

- The DNS lists what you think you have
- What do you *really* have?
- You can ping IP addresses

```bash
for i in `seq 1 254`
do
    ping 128.59.23.$i
done
```
How About a Broadcast ping?

# ping -L -r -w 100 128.59.23.255
PING 23-net.cs.columbia.edu (128.59.23.255): 56 data bytes
64 bytes from 128.59.18.102: icmp_seq=0 ttl=255 time=3.848
64 bytes from 128.59.20.155: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.22.252: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.22.7: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.18.134: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.22.7: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.18.133: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.18.134: icmp_seq=0 DUP! ttl=64
64 bytes from 128.59.22.7: icmp_seq=0 DUP! ttl=64
time=6.505
# ping -L -r -w 100 128.59.23.255
PING 23-net.cs.columbia.edu (128.59.23.255): 56 data bytes
ping: sendto: Network is unreachable

- “Directed broadcasts” are blocked to prevent *Smurf* attacks
- Smurf attack: send a ping packet to a broadcast address, with the (forged) source address of your victim
- Many hosts will send back to it, using up lots of the victim’s bandwidth
ARP

- If we’re on the same LAN, we can learn more via ARP:

```
# arp -a
mudd-edge-1.net.columbia.edu (128.59.16.1) at 00:
dynasty.cs.columbia.edu (128.59.16.5) at 00:03:
```

```
disco.cs.columbia.edu (128.59.16.7) at 08:00:20:
razor.cs.columbia.edu (128.59.16.8) at 00:01:02:
```

- Note that the first three bytes of the MAC address tell who manufactured the card: 00:d0:06 is Cisco, 00:03:ba and 08:00:20 are Sun, etc.
NMAP

The Basics

Finding Hosts

Port-Scanning

The Real Truth About CS...

Trying it From Home

Mapping Versions

Local Software

Learning Versions

To Tell the Truth?

Fingerprinting

Evasive Action

Conclusions
The Network Map Tool

- General-purpose scanner
- Does everything I’ve described and more
- Practically point-and-click scanning (but it’s command-line)
Finding Hosts

# nmap -sP 128.59.23.0/21
Host mudd-edge-1.net.columbia.edu (128.59.16.1) appears to
Host dynasty.cs.columbia.edu (128.59.16.5) appears to
Host mailswitch.cs.columbia.edu (128.59.16.6) appears to
Host disco.cs.columbia.edu (128.59.16.7) appears to
Host razor.cs.columbia.edu (128.59.16.8) appears to
...
# nmap -sP 128.59.23.0/21
Host mudd-edge-1.net.columbia.edu (128.59.16.1) appears to be up
MAC Address: 00:D0:06:26:9C:00 (Cisco Systems)
Host dynasty.cs.columbia.edu (128.59.16.5) appears to be up
MAC Address: 00:03:BA:14:A3:68 (Sun Microsystems)
Host mailswitch.cs.columbia.edu (128.59.16.6) appears to be up
MAC Address: 00:17:08:B5:41:00 (Hewlett Packard)

...
Port-Scanning

- Find out what ports are open on a machine
- Better yet, find out what applications are behind those ports
- Extras: avoid detecting, detect firewalls, bypass some firewalls, etc.
The Real Truth About CS...

# nmap -p 1-200 cs.columbia.edu

Not shown: 195 closed ports

<table>
<thead>
<tr>
<th>PORT</th>
<th>STATE</th>
<th>SERVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>22/tcp</td>
<td>open</td>
<td>ssh</td>
</tr>
<tr>
<td>25/tcp</td>
<td>open</td>
<td>smtp</td>
</tr>
<tr>
<td>53/tcp</td>
<td>open</td>
<td>domain</td>
</tr>
<tr>
<td>111/tcp</td>
<td>open</td>
<td>rpcbind</td>
</tr>
<tr>
<td>139/tcp</td>
<td>open</td>
<td>netbios-ssn</td>
</tr>
</tbody>
</table>

MAC Address: 00:03:BA:62:6A:39 (Sun Microsystems)

Nmap finished: 1 IP address (1 host up) scanned in 6.249 seconds

Many fewer ports than in the WKS record...
7/tcp filtered echo
9/tcp filtered discard
19/tcp filtered chargen
22/tcp open ssh
25/tcp open smtp
53/tcp open domain
111/tcp open rpcbind
135/tcp filtered msrpc
136/tcp filtered profile
137/tcp filtered netbios-ns
138/tcp filtered netbios-dgm
139/tcp filtered netbios-ssn
# nmap -sA -p 1-200 www.cs.columbia.edu

<table>
<thead>
<tr>
<th>PORT</th>
<th>STATE</th>
<th>SERVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>135/tcp</td>
<td>filtered</td>
<td>msrpc</td>
</tr>
</tbody>
</table>
Sometimes It’s Like This

```
3/tcp  filtered compressnet
7/tcp  filtered echo
36/tcp filtered unknown
116/tcp filtered ansanotify
132/tcp filtered cisco-sys
135/tcp filtered msrpc
147/tcp filtered iso-ip
157/tcp filtered knet-cmp
177/tcp filtered xdmcp
```

Different paths? Or a scan failure? Unclear.
How does nmap detect a filtered service?
- A TCP SYN is normally answered with a SYN+ACK or a RST
- A filtered port generally returns nothing
ACK Scans

- Send a packet with the ACK bit set
- Gets through packet filters!
- Can’t distinguish between open and closed services; can be used to map firewall rules
Avoiding Detection

- If a program does a `connect()` call, the usual 3-way TCP handshake will occur.
- The application can log the fact and source of the connection.
- `Nmap` hand-crafts SYN packets, and responds to any SYN+ACK with RST.
- The TCP open never completes, so the application never notices and can't log.
UDP Ports

- Send a UDP packet
- Watch for a response or an ICMP Port Unreachable
- No answer at all may indicate a filtered port
Why do we want to?

- Particular applications may have (security) bugs
- Particular versions of particular applications may have (security) bugs

Starting Nmap 4.11 ( http://www.insecure.org/nmap/ )

Interesting ports on shadow.cs.columbia.edu (128.59.23.100):

Not shown: 196 closed ports

<table>
<thead>
<tr>
<th>PORT</th>
<th>STATE</th>
<th>SERVICE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>22/tcp</td>
<td>open</td>
<td>ssh</td>
<td>OpenSSH 3.9p1 (protocol 1.99)</td>
</tr>
<tr>
<td>25/tcp</td>
<td>open</td>
<td>smtp</td>
<td>Sendmail 8.12.10/8.12.10</td>
</tr>
<tr>
<td>80/tcp</td>
<td>open</td>
<td>http</td>
<td>Apache httpd 1.3.33 ((Unix) mod_ssl/2.8.22)</td>
</tr>
<tr>
<td>111/tcp</td>
<td>open</td>
<td>rpcbind</td>
<td>2-4 (rpc #100000)</td>
</tr>
</tbody>
</table>

MAC Address: 00:03:BA:C5:A0:DD (Sun Microsystems)

Device type: general purpose

Running: Sun Solaris 8

OS details: Sun Solaris 8

Uptime 13.412 days (since Thu Oct 19 15:52:13 2006)

Service Info: OS: Unix
How does nmap get that data?

Many services announce it right away:

```
# telnet www.cs.columbia.edu 80
Trying 128.59.23.100...
Connected to shadow.cs.columbia.edu.
Escape character is '^[].'
GET / HTTP/1.0
```

HTTP/1.1 200 OK
Date: Thu, 02 Nov 2006 05:49:38 GMT
Server: Apache/1.3.33 (Unix) mod_ssl/2.8.22 OpenSSL/0.9.7e
X-Powered-By: PHP/4.3.11

In other cases, it uses heuristics
$ dig version.bind txt chaos @kedu.cc.columbia.edu
version.bind. 0 CH TXT "9.2.6-P1"

$ dig version.bind txt chaos @cs.columbia.edu
VERSION.BIND. 0 CH TXT "surely you must be joking"

Hiding the version helps less than you might think
Various heuristics can be used to identify OS and version

Example: look at initial sequence number patterns, support for TCP options, initial window size, etc.

Get uptime from TCP timestamp option

Evaluate sequence number and IPid field predictability

But good guys need version numbers for site management

Net result: hiding version numbers tends to hurt the good guys more than the bad guys
Evasive Action

- Nmap has many techniques to avoid detection
- Example: randomized scan orders, decoy hosts, zombies, bounce attacks, etc.
- Nasty example: --badsum
- Send packet with a bad TCP checksum
- Hosts will drop such packets — but some IDS won’t
Conclusions

- Scanning is a very powerful attack technique.
- It’s very hard to hide from a clever scanning program.