Global Information Assurance Certification Paper

What is on your Intranet?

Sunil D. Shah
February 20, 2001

Introduction

The explosive growth of the Internet since the mid-nineties has been paralleled to a large extent by the growth of intranets within companies to enable information sharing, streamline business processes and so on. Intranets utilize the same technologies as the Internet; the main distinction is that intranets are contained within an enterprise and are intended for use by people inside the company. Intranets are often connected to the public Internet, typically with a firewall to provide a first line of defense.

It is, indeed, a rare company that has implemented an intranet per a master plan! In a majority of the cases, the intranet just happened (seemingly overnight). Thus, at some point, businesses have to pause and ask the question, "What is on our intranet?" This question is important due to a variety of reasons including data confidentiality, improper (vulnerable) setups of web servers, the threat from insiders, and the growing presence of non-employees such as leased workers, alliance partners, suppliers, and consultants on a company's networks. Getting the answer to this question is a pre-requisite to follow-up steps aimed at improving the security of company information on the intranet.

The Search for Web Servers

One approach to identifying your web servers is to map the network using a security scanner. There are a number of very capable scanners available, both commercial (*ISS Internet Scanner, Cisco Secure Scanner, Axent NetRecon* et al.) and freeware/open source (*Nmap, Nessus, Netcat, SATAN, Whisker*, and others)[1]. Features vary from product to product but commonly include identification of active hosts, network mapping, port scanning, and operating system identification. The popular *Nmap* scanner (available at http://www.insecure.org) will be used for the purposes of illustration in this paper.

Parlez-vous HTTP?

The problem of identifying web servers boils down to identifying the subset of nodes on your intranet that have active HTTP ports. Running a comprehensive scan on the address range(s) of your network looking for HTTP in the results is not advised since it will generate heavy traffic, likely break something, produce large volumes of data, and take a considerable length of time. Rather, you can reduce the traffic and speed up the process by limiting the scan to the few standard port numbers assigned to HTTP. These can be obtained from the list maintained by the *Internet Assigned Numbers Authority (IANA)*[2]:

```
http            80/tcp    World Wide Web HTTP
http            80/udp    World Wide Web HTTP
https          443/tcp    http protocol over TLS/SSL
https          443/udp    http protocol over TLS/SSL
http-alt      8080/tcp    HTTP Alternate (see port 80)
http-alt      8080/udp    HTTP Alternate (see port 80)
```

Limiting the scan of the network, then, to the three ports 80, 443, and 8080 (instead of thousands for a "full" scan) should identify all the active web servers on your intranet that are using the standard HTTP ports. (If your company has decided to use additional non-standard ports for some reason, you should include them in the scan).

> **CAUTION:** Before proceeding any further, it should be noted that network scanning can be very hazardous to your career[3]. Nmap and other scanners are not designed to be denial of service tools but the problems they cause from time to time can be surprising. Familiarize yourself with the operation of the scanner and obtain authorization (written, if appropriate) from your supervisor or manager. Do not run a scan unattended, make it as narrow/non-invasive as possible, and ensure that people know what you are doing and how to contact you.

So now, one should be set to run the scan for the HTTP ports on the network. An example of a command line for nmap would be:

```
nmap –p 80,443,8080 128.210.3.1-7
```

This will scan the first 7 addresses on subnet 3 of the 128.210 class 'B' address space for ports 80, 443, and 8080. You should change the addresses to match your network. Nmap offers considerable flexibility in specifying address ranges; details are available in the man page[4] for nmap that serves as its primary documentation. The above command will generate output to your screen that might look like the following:

```
Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/
)
All 3 scanned ports on node1.company.com (128.210.3.1) are: closed
Interesting ports on node2.company.com (128.210.3.2):
(The 2 ports scanned but not shown below are in state: closed)
Port       State       Service
80/tcp     open        http

All 3 scanned ports on node3.company.com (128.210.3.3) are: closed
Interesting ports on node4.company.com (128.210.3.4):
(The 1 port scanned but not shown below is in state: closed)
Port       State       Service
80/tcp     open        http
443/tcp    open        https

All 3 scanned ports on node6.company.com (128.210.3.6) are: closed
All 3 scanned ports on node7.company.com (128.210.3.7) are: closed
```

```
Nmap run completed -- 7 IP addresses (6 hosts up) scanned in 2
seconds
```

As you can see, nmap quickly scanned the specified range of 7 addresses and found 6 hosts, two of which are running HTTP (*node2* and *node 4*) and, presumably, are web servers.

Scaling up the Search

To identify the web servers on your network, then, you need to expand the range of addresses to cover the extent of your network. If your network is extensive, you may want to consider running multiple nmap scans, each covering a manageable subset of your address space. This approach will help keep scan times down to reasonable levels (remember, you will be attending to the scan waiting for it to finish), and will reduce the need to start over from scratch should a scan hang up.

Another issue to consider as you scale up is the capture and analysis of the information reported by nmap. While one could certainly redirect the *human readable* output of the command (illustrated in the previous section) to a file, the subsequent analysis is likely to be tedious. Fortunately, nmap offers the option to log the results of the scan in a *machine parseable* form to a file you specify as an argument:

```
nmap -p 80,443,8080 -oM n_out 128.210.3.1-7
```

This will scan the same 7 nodes we listed earlier except that machine parseable output will be generated and saved in the file **`n_out`** (specified by the option **`-oM n_out`**). The file **`n_out`** will contain: (4 of the lines below are long and have wrapped)

```
# Nmap (V. nmap) scan initiated 2.53 as: nmap -p 80,443,8080 -oM
n_out 128.210.3.1-7
Host: 128.210.3.1 (node1.company.com)    Status: Up
Host: 128.210.3.2 (node2.company.com)    Ports: 80/open/tcp//http///
      Ignored State: closed (2)
Host: 128.210.3.3 (node3.company.com)    Status: Up
Host: 128.210.3.4 (node4.company.com)    Ports: 80/open/tcp//http///,
443/open/tcp//https/// Ignored State: closed (1)
Host: 128.210.3.6 (node6.company.com)    Status: Up
Host: 128.210.3.7 (node7.company.com)    Status: Up
# Nmap run completed at Mon Feb 19 09:10:07 2001 -- 7 IP addresses (6
hosts up) scanned in 1 second
```

There are several differences above from the human readable output (illustrated in the previous section) that can greatly ease the data analysis task:

- The first line documents the exact command used to generate the report.
- A one-line record is generated for each active host identified in the scan. This makes it very easy to quickly perform a variety of analyses to suit your needs. For example, the UNIX command **`fgrep http n_out | wc -l`** would yield the count (=2) of web servers running http (i.e. *node 2* and *node 4*). Also, the

command **`fgrep http n_out | cut -f2 -d' ' > web_addrs`** would generate a list of the TCP/IP addresses (one per line) for each of the nodes (2 in the case above) found running http, and save it to a file **`web_addrs`**.

- The last line documents the date and time that the scan was run.

Believe it or not

When you complete scanning your entire address space and compiling a summary of the results, you should not be surprised at all if you find an unbelievably large number of web servers on your network. This warrants further investigation, and, now that you have the TCP/IP addresses of all these servers, you can simply point your browser to some of these addresses to find out what is going on. What you are likely to find is a variety of network devices that have "embedded" web servers to allow management tasks to be performed through direct communication with a web browser:

- Printers
- Copiers
- FAX machines
- PBXs
- Uninterruptible power supplies
- Remote server management interfaces
- Routers, hubs, and switches
- …..

While these devices are interesting and have their own sets of potential vulnerabilities[5], they divert us from our primary purpose – that of identifying the "real" web servers, ones that serve up company *information*. The effort so far has not been for naught, though, since you have succeeded in whittling down your entire address space to a list of hosts running HTTP (saved in the file **`web_addrs`**).

Will the "real" web servers please stand up?

While the brute force method of visiting each of the identified web server addresses to pick out "real" web servers will certainly work, it is likely to be very laborious on networks of any appreciable size. An automated method is clearly desirable. One approach involves the determination of the operating system (OS) running on each of the web servers. Once the OS is known, you can pick out the "real" web servers by looking for "reasonable" operating systems such as UNIX, Windows, OpenVMS, etc. The nmap scanner has strong TCP/IP fingerprinting capabilities for OS detection that can be used to advantage here. Details on the technique employed are beyond the scope of this paper but are available in an excellent article by the author of nmap[6]. You can ask nmap to identify the operating system for *node2* using the **`–O`** option:

```
nmap –O node2
```

The output might look like the following:

```
Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/
)
Interesting ports on node2.company.com (128.210.3.2):
(The 1519 ports scanned but not shown below are in state: closed)
Port       State       Service
80/tcp     open        http
135/tcp    open        loc-srv
512/tcp    open        exec
1433/tcp   open        ms-sql-s

TCP Sequence Prediction: Class=trivial time dependency
                         Difficulty=6 (Trivial joke)
Remote operating system guess: Windows NT4 / Win95 / Win98

Nmap run completed -- 1 IP address (1 host up) scanned in 1 second
```

Nmap easily identified *node2* as being a Windows machine (see the second to last line in the sample output above). It should be noted that while nmap's extensive database of TCP fingerprints enables it to detect well over 400 operating systems, it does have limitations. When it encounters an unfamiliar device, it will generate the message, "`No OS matches for host`". This should not pose a significant problem for the task at hand, though, since one expects real web servers to be hosted on reasonably well-known operating systems.

Building on our example so far, we can ask nmap to identify the operating system for each of the web servers we found earlier:

`nmap -O -iL web_addrs -oM os_out`

The **`-iL`** option tells nmap to read a list of target addresses from the file **`web_addrs`** that we generated in an earlier example. It contains the addresses of *node2* and *node4*:

```
128.210.3.2
128.210.3.4
```

As before, the **`-oM`** option is used to specify machine parseable output; the results are saved in a file **`os_out`** that may look something like the following:

```
# Nmap (V. nmap) scan initiated 2.53 as: nmap -O -iL web_addrs -oM o
s_out
Host: 128.210.3.2 (node2.company.com)   Ports: 80/open/tcp//http///,
 135/open/tcp//loc-srv///, 512/open/tcp//exec///, 1433/open/tcp//ms-
sql-s///        Ignored State: closed (1519)    Seq Index: 4    OS:
Windows NT4 / Win95 / Win98
Host: 128.210.3.4 (node4.company.com)   Ports: 21/open/tcp//ftp///,
23/open/tcp//telnet///, 80/open/tcp//http///, 111/open/tcp//sunrpc//
/, 443/open/tcp//https///, 512/open/tcp//exec///        Ignored Stat
e: closed (1517)        Seq Index: 1    OS: HP-UX 10.20 E 9000/777 o
r A 712/60 with tcp_random_seq = 0
# Nmap run completed at Mon Feb 19 16:29:02 2001 -- 2 IP addresses (
2 hosts up) scanned in 1 second
```

Again, you have one (rather long) line per host, and nmap reports its guess for the OS towards the end of the line. Given the convenient tab-delimited format of the data in the **os_out** file, you can look for "reasonable" operating system names to pick out the real web servers. Before proceeding to that step, it is helpful to generate a list of all operating systems identified by nmap in the **os_out** file for your environment as follows:

```
fgrep Host: os_out | cut –f5 | sort | uniq
```

This UNIX command uses **fgrep** to select lines containing the string **Host:** and passes the output to **cut** to extract the OS name (field 5). **sort** and **uniq** are then used to eliminate duplicate entries from the sorted output which, in our simple example, would be as follows:

```
OS: HP-UX 10.20 E 9000/777 or A 712/60 with tcp_random_seq = 0
OS: Windows NT4 / Win95 / Win98
```

You can now select the operating systems that you deem "reasonable" for hosting real web servers and run queries for each of them as follows:

```
fgrep Windows os_out | cut -f1,5
```

This UNIX command line uses **fgrep** to extract all lines containing the nmap string "Windows" from the file **os_out**. These lines are piped to **cut** which extracts the host address and associated OS name (fields 1 and 5) to yield:

```
Host: 128.210.3.2 (node2.company.com)   OS: Windows NT4 / Win95 /
Win98
```

Similar queries can be run for all other operating systems of interest.

Conclusion

The answer to the question, "What is on our intranet?" is not an easy one to answer for company networks of any appreciable size. This paper has highlighted some of the key issues to be considered and has presented an automated methodology for remotely identifying "real" web servers (*information* servers) on a large network with a high degree of accuracy. This paves the way for follow up efforts to ensure adequate security for company information on the intranet.

---

[1]Black, Ronald. "How Does Network Security Scanning Work Anyway?" 10 Sep 2000
URL: http://www.sans.org/infosecFAQ/securitybasics/netsec_scanning.htm (16 Feb 2001)

[2] IANA Port Numbers
URL: http://www.isi.edu/in-notes/iana/assignments/port-numbers (17 Feb 2001)

[3] Northcutt, Stephen. "Internet Threat Brief" SANS Security Essentials Curriculum. V1.1. 11 Feb 2000

(20 Oct 2000)

[4] Fyodor. “Nmap network security scanner man page”. 2000
URL: http://www.insecure.org/nmap/nmap_manpage.html (23 Jan 2001)

[5] Smith, Kevin. “Do You Copy? Security Issues with Digital Copiers.” 16 Sep 2000
URL: http://www.sans.org/infosecFAQ/threats/copy.htm (16 Feb 2001)

[6] Fyodor. "Remote OS detection via TCP/IP Stack FingerPrinting." 10 Apr 1999.
URL: http://www.insecure.org/nmap/nmap-fingerprinting-article.html (17 Feb 2001)