OS Fingerprinting Techniques & Tools

An exposition by

Steve Humer & Ashley Murphy

CS-455: Cryptography & Network Security
Keene State College, November 2013
## Contents

Abstract .......................................................................................................................... 2

1. Introduction .................................................................................................................. 3
   1.1. What is OS fingerprinting? ...................................................................................... 3
   1.2. Why OS fingerprinting? .......................................................................................... 3

2. Techniques .................................................................................................................... 4
   2.1. Non-Automated Techniques .................................................................................. 4
       2.1.1. Direct & Indirect Banner Grabbing ................................................................. 4
       2.1.2. TCP & ICMP Fingerprinting ............................................................................. 5
       2.1.3. Active & Passive fingerprinting........................................................................ 5
   2.2. Automated Techniques ........................................................................................... 6

3. Tools ............................................................................................................................. 7
   3.1. nmap ..................................................................................................................... 7
   3.2. Xprobe2 ................................................................................................................ 8
   3.3. Wireshark .............................................................................................................. 9

4. Perspective ................................................................................................................... 10
   4.1. Internal Uses ...................................................................................................... 10
   4.2. External Threats .................................................................................................. 10

Bibliography ................................................................................................................... 11
Abstract

This exposition will supply insight into the process behind various OS fingerprinting techniques and uses. To begin, OS fingerprinting is the process of identifying a remote host’s operating system based on analyzing packets from that host. OS fingerprinting is useful for both Network Administrators to maintain or improve a system and hackers looking to disrupt or find fault with a system. Non-automated techniques involve three distinct groups. Direct and indirect banner grabbing may grab OS data since hosts often announce their OS to anyone making a connection to them through banners. TCP and ICMP fingerprinting techniques may discover OS data through the protocol connections. Active scanners, which send data before analyzing packets, and passive sniffers, which analyze packets alone, may option crucial information. Automated techniques have gained fame, notable active scanners are nmap and Xprobe2, leading to a dissection of their process and uses. The automated passive sniffer p0f is notable, but not discussed; a demonstration in Wireshark shows passive sniffing manually. Further, internal uses and external threats in an increasingly tech-savvy world of three users, Network Administrators, ethical hackers, and malicious hackers, will be discussed as we move forward.
Section 1

Introduction

1.1. What is OS fingerprinting?

Understanding OS fingerprinting first involves an understanding of the terminology behind the name. A fingerprint, in simple terms, is a unique way to identify a human being. Similarly, an Operating System (OS) has unique characteristics in its communication implementation that allow it to be identified on a network.

Through the process of analyzing certain protocol flags, options, & data in the packets a device sends onto the network, we can make relatively accurate guesses about the OS that sent those packets. Wikipedia\(^1\) states OS fingerprinting as “the process in computing of determining the identity of a remote host’s operating system by analyzing packets from that host.”

1.2. Why OS fingerprinting?

OS fingerprinting has both internal uses and external threats associated with it. By utilizing OS fingerprinting, one may improve system maintenance, efficiency, and protection, secure the system against threats to these aspects, or possibly engage in threatening activities themselves.

“Network scanning, and particularly remote OS/application detection, is generally the first step in mapping out a network; whether for penetration testing or simply maintaining a network device inventory.” (Allen) For example, a Network Administrator may run a scan on his own network and know all the devices present. By periodically running the same scan, they could keep their network inventory clean by knowing when new devices were installed and old devices were disused.

On the other hand, when an attacker is trying to hack into any computer, he starts to gather information about the computer (target) as much as possible. (Nawaratne) The operating system the target is running on is key information. An ethical hacker who uses OS fingerprinting techniques may work to protect a system from vulnerabilities, while a malicious hacker who finds a vulnerable system may gain access to OS information and plan an attack.

Section 2

Techniques

2.1. Non-Automated Techniques

There are three identifiable groupings of non-automated techniques, all of which may help successfully obtain a host’s OS fingerprint without the aid of third-party applications. If one technique fails, another may be used or a combination of multiple techniques may be used to obtain all the necessary information. Many of these techniques do not directly obtain a host OS, but obtain host server, router, or miscellaneous device information that leads to discovering the host OS fingerprint.

The three groupings are split into two sub-groups, with each being discussed below. The context of the situation will likely determine which sub-group is more ideal for different scenarios.

2.1.1. Direct & Indirect Banner Grabbing

The simplest and easiest OS fingerprinting technique involves direct and indirect banner grabbing. “Hosts will often announce their OS to anyone making a connection to them through welcome banners or header information.” (Nawaratne) A few different types of software, including Telnet and various proprietary programs, can be used to perform banner grabbing.

Banners are grabbed by connecting to a host, and then sending a request to a port that is associated with a particular service, such as port 80 for HTTP. The banner for HTTP will typically show the type of server software, version number, and other similar information. Once an attacker has access to software information, they can find an exploit for that specific version. For example, a Microsoft Exchange email server would only be installed on a Windows OS.

Network Administrators may use banner grabbing for HTTP fingerprinting, performing an inventory on all of the different services and systems operating on the host, and other activities. Malicious hackers will use banner grabbing when looking for vulnerable hosts. They will establish a connection with a host, and then query ports looking for vulnerable services. “OS fingerprinting in the majority of cases is always a combination of the output from as many probes/tools as possible, so faking just one banner output will not be enough.” (wisegeek)
2.1.2. TCP & ICMP Fingerprinting

To understand TCP fingerprinting requires a look at a TCP header, as shown:

```
+----+-----+----+----+----+----+----+----+
|   4 |   8 |  12|  16|  20|  24|  28|  32|
+----+-----+----+----+----+----+----+----+
|     |     |    |    |    |    |    |    |
+----+-----+----+----+----+----+----+----+
| Sequence Number | Acknowledgement Number |
| Offset | Reserved | Flags | Window Size |
| Checksum | Urgent Pointer |
+----+-----+----+----+----+----+----+----+
|     |     |    |    |    |    |    |    |
+----+-----+----+----+----+----+----+----+
| Options | Padding |
+----+-----+----+----+----+----+----+----+
| Data |
```

*Figure 2.1: TCP header fields [Hun92]*

“The flags header fields are essential for OS fingerprinting since each operating system reacts differently to normal and special crafted TCP packets sent to its network stack.” (Nostromo) When two members first establish a connection, each computer selects an ISN (Initial Sequence Number) and each subsequent packet will be numbered by counting up from that number. “By analyzing the ISNs generated by a target, a scanner can possibly determine, or at least narrow down the possibilities of, the target OS.” (Allen) The addition of TCP options, including Timestamps, Window Scaling, Maximum Segment Size, and Explicit Congestion Notification (ECN) may immediately reveal a target OS.

To understand ICMP fingerprinting requires a look at an ICMP header, as shown:

```
+----+-----+----+----+----+----+----+----+
|   4 |   8 |  12|  16|  20|  24|  28|  32|
+----+-----+----+----+----+----+----+----+
| Type | Code |    |    |    |    |    |    |
+----+-----+----+----+----+----+----+----+
| Checksum |
+----+-----+----+----+----+----+----+----+
| Data |
```

*Figure 2.2: ICMP header fields [Hun92]*

The Type field specifies the format of the ICMP message, and the header looks different for each request and reply packet. Essentially, each host has an individual ICMP fingerprint. “Network Administrators use ICMP daily to monitor the status of servers or routers, using a simple ICMP ping.” (Allen)

2.1.3. Active & Passive Fingerprinting

Active fingerprinting involves scanning and works by sending a series of specially crafted packets to the target host and analyzing the replies. This allows the scanner to obtain more accurate results than a passive scanner and in a shorter amount of time. (Allen) The responses are then compared to a database to determine the operating system. However, this type of fingerprinting is detectable because it repeatedly attempts to connect with the same target system. (Understanding banner grabbing)
Passive fingerprinting involves sniffing and does not generate any traffic during the fingerprinting session, but instead analyze existing traffic between the scanning host and the target. Passive fingerprinting can be done completely offline, by examining packets captured previously. Passive scanners are generally and inherently less accurate than active scanners, due to the fact they have less control over the data they are analyzing. (Sans)

2.2. **Automated Techniques**

The non-automated techniques discussed in section 2.1 may, and successfully have, been converted into automated third-party applications. A discussion of these tools will reiterate the techniques above, and further extend the uses of OS fingerprinting.

The following sub-sections will discuss the use of nmap for active scanning through the use of ICMP/TCP packets, and Xprobe2 for active scanning relying primarily on ICMP techniques. A passive scanner, p0f, will not be discussed, but is a great alternative if stealth operation is important in a particular scenario. Further, we can use Wireshark to capture a packet and run commands to identify the host OS, similar to a passive scanner.
Section 3

Tools

3.1. nmap

“nmap (‘Network Mapper’) is an open source utility for network exploration or security auditing.” (nmap, Introduction P1) The primary purpose of nmap is to rapidly scan large segments of a network for devices that are active, with the ability to report which ports are open on those devices. nmap does this through active scanning and it is arguably one of the most popular network scanners in use today. (Allen)

A simplified understanding of nmap’s method is useful to identify with nmap’s process. “nmap begins its OS detection by sending an ICMP ping request to the target, then it connects to port 80 (HTTP) to see if the target is responding and running at all. Then nmap does the actual portscan, searching for at least one open and one closed port. At least four packets are sent to an opened port, and three to a closed port, with the replies recorded. Afterward, a lookup in the OS-detection fingerprint file is made. (Nostromo) TCP is mainly used in conjunction with ICMP, UDP and probes to reach these results. By utilizing several methods, nmap is able to search continuously until finding interesting and useful ports that are open to obtain data.

Thus when running nmap, identified OS details may result in the following display:

As seen, nmap is able to identify aggressive OS guesses for a specific target, i.e. Linux 2.6.32.
3.2. Xprobe2

Unlike nmap, Xprobe2 does not work with TCP packets, but with ICMP packets to collect the data needed for OS fingerprinting. Xprobe2 can be especially useful when there are no open ports available on the target device. It uses a matching system based on calculations to prevent false detections when something is between the scanner and the target.

When Xprobe2 completes an active scan of a target, the scores for each test are compared to the signature database and a probable OS is determined by the database entry which most closely resembles the results. (Allen) The interesting thing is that Xprobe2 has not had active development done since 2005, thus showing the robust role of a strong ICMP scanner.

Below is an example scan using Xprobe2:

As seen, Xprobe2 is able to identify the OS with much certainty, i.e. Linux Kernel 2.4.22.
3.3. Wireshark

With active techniques aside, passive techniques can be an easy way of doing OS fingerprinting. By simply looking at the TCP window size and Time To Live (TTL) in the IP header of the first packet in a TCP session. A table showing the values for popular operating systems is necessary.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Time To Live</th>
<th>TCP Window Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux (Kernel 2.4 and 2.6)</td>
<td>64</td>
<td>5840</td>
</tr>
<tr>
<td>Google Linux</td>
<td>64</td>
<td>5720</td>
</tr>
<tr>
<td>FreeBSD</td>
<td>64</td>
<td>65535</td>
</tr>
<tr>
<td>Windows XP</td>
<td>128</td>
<td>65535</td>
</tr>
<tr>
<td>Windows Vista and 7 (Server 2008)</td>
<td>128</td>
<td>8192</td>
</tr>
<tr>
<td>iOS 12.4 (Cisco Routers)</td>
<td>255</td>
<td>4128</td>
</tr>
</tbody>
</table>

By using tshark, a file installed along with Wireshark, we can obtain all the SYN packets after capturing packets with Wireshark while connecting to a website.

![Image of tshark output](image)

In this instance, my IP address is 158.65.227.44 with a TTL of 128 and a TCP Window Size of 8192. By comparing this data with the chart above, we can see that my machine is likely a Windows Vista or Windows 7 operating system. Accurately, my computer runs Windows 7.
Section 4
Perspective

4.1. Internal Uses
As discussed above, the internal uses of OS fingerprinting are clear. Network Administrators need, and should, be able to view everything on the network they are maintaining. This is even more essential moving forward, as more and more people are connecting technology to networks. The cell-phone movement means everyone has a smart phone that can connect to a network, WiFi has become an essential custom, cameras, watches, and various devices are straining our systems even more. Moore’s law states that the transistor count doubles every two years, with technology constantly being outdated and in need of updates.

Controlling the massive influx of technology, with the massive need to stay up-to-date in a forward thinking society requires OS fingerprinting to help stay in stride. The ability to read device information is crucial, and should be necessary material to learn for anyone in the networking business. With global IPv6 addresses replacing IPv4, the limits of today will be replaced by the limitless OS fingerprinting opportunities of tomorrow.

4.2. External Threats
On the other side of the equation are the ethical and malicious hackers, who are also growing in numbers. As the technological field increases, and kids become highly invested in technology at an early age, the boom moving forward is bound to be great. There will always be both sides of the hacking equation, ethical and malicious, with the real battle being between one upping one another. Ethical hackers will have to work to protect systems before malicious hackers find ways to override those protective measures.

This ensuing battle will continue to mean that information related to system protection, especially as societies rely more and more on technology, will become an even higher priority. So far, the general trend toward increasing penalties for getting caught has led to the increased trend in worm toolkits being developed to refine attacks. It is not just wise, but essential, for us to have a familiar understanding of these techniques and the systems we are involved in as a result in such a connected environment.
Bibliography


