Chapter 19

PASSIVE DETECTION OF NAT ROUTERS AND CLIENT COUNTING

Kenneth Straka and Gavin Manes

Abstract Network Address Translation (NAT) routers pose challenges to individuals and organizations attempting to keep untrusted hosts off their networks, especially with the proliferation of wireless NAT routers. Residential NAT routers also create problems for Internet Service Provider (ISP) taps by law enforcement by concealing network clients behind cable or DSL modems. This paper discusses the feasibility and limitations of methods for detecting NAT routers and counting the number of clients behind NAT routers.

Keywords: NAT routers, passive detection, client counting

1. Introduction

Network Address Translation (NAT) routers provide a convenient way to share a single IP address between several clients, but they can pose serious security risks [3, 8]. A NAT router connected to an Ethernet port in an office environment allows several—possibly untrusted—clients to access network resources. Some ISPs use MAC address filtering to keep multiple clients from using their services. However, commercial NAT routers can clone a client's MAC address. This MAC cloning allows a NAT router to masquerade as a client computer so that the connected network does not suspect other connected devices. In this sense, a NAT router can subvert many commonly used security protocols, e.g., MAC address authentication. Therefore, MAC address filtering is not an effective technique for detecting and mitigating NAT routers.

NAT router functionality is commonly combined with IEEE wireless fidelity standards 802.11b and 802.11g [2], allowing users with appropriately equipped computers or laptops to wirelessly access the Internet via a NAT router. Users without specialized training may not realize that

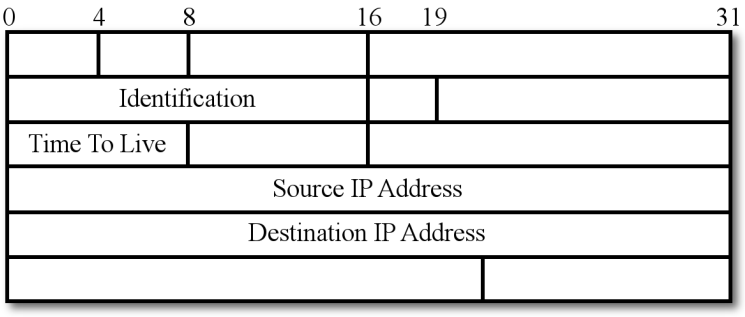

Figure 1. IP header fields relevant to detecting NAT routers and counting clients.

plugging a wireless NAT router into an Ethernet port with default settings in order to use a wireless laptop can create an insecure access point to the internal network [7]. A war-driver or other malicious entity outside the physical protection perimeter could access the internal network and launch an attack [4]. Finding and incapacitating rogue wireless NAT routers consume time and resources—detection usually involves "war-walking" with a wireless laptop to search for unauthorized wireless access points. Thus, it is often the case that rogue routers remain operational for extended periods of time.

In addition to leaving a network open to unauthorized clients, NAT routers effectively conceal the identity and number of connected clients. Therefore, ISP taps performed by law enforcement in computer crime investigations may not provide accurate information. Knowing the number and type of clients residing behind a router is useful for preparing a search warrant, as it may indicate the number of network clients to be seized and the level of technological sophistication of the suspect. Furthermore, detecting unauthorized NAT routers is essential to maintaining security in any enterprise network.

A NAT router shares a single Internet connection between several network clients by assigning a local, non-routable IP address to each network client [8]. The NAT router acts as a gateway when a client sends a packet beyond the NATed LAN (a remote server). When the NAT router receives the packet, it changes several fields (Figure 1). The most important is the packet's source IP address, which is replaced with that of the router. This allows the response from the remote server to return to the correct world-routable IP address. In addition, the NAT router replaces the packet's original source port with an arbitrary high-numbered port associated with the client's source port. Upon receiving the response from the remote server, the NAT router checks the destina-

tion port against its table of client source ports. It then routes the packet to the appropriate client after recreating the original source port. This is necessary because a packet received by the router is addressed to the world-routable IP address, and without an additional demux key (the destination port on the inbound packet) it is unclear which client should receive the packet. When a new source port is used by any client for the next stream, it also translates to a new source port on the post-NAT router packet (usually the last translated source port plus one, regardless of which client created the new source port), showing a one-to-one relationship between original source ports and translated source ports. The source port progression apparent on the public side of the NAT router is almost indistinguishable from that of a directly-connected, solitary client.

The next section describes current methods for detecting NAT routers. The remaining sections discuss the feasibility and limitations of methods for detecting NAT routers and counting the number of clients.

2. Detection Methods

Although several methods exist to actively detect access points and NAT routers, they are less than ideal. The Network Mapper (nmap) can be used to perform TCP fingerprinting on network clients and determine which clients are running a router OS [5, 6]. This process, which is prone to false positives and false negatives, may take a long time for a large network and produces suspicious traffic that can trigger a local intrusion detection system (not a good attribute for a network defense tool). Another active method is similar in its approach and flaws. It uses Simple Network Management Protocol (SNMP) [10] scanning in combination with the `snmpwalk` tool to determine a client's OS [9].

Several methods exist to passively detect NAT routers. Examining the TTL (time to live) value of an IP packet can help determine if the packet has passed through a network device (NAT router) that decremented the TTL. In this case, the TTL value is one less than what is expected when the packet reaches the internal networking hardware.

Another method for detecting NAT routers and counting clients is to examine the id field in IP packet headers; this field is often used as an unofficial counter of outbound packets [1]. However, this method can result in false positives because some operating systems use the per-source-port counter as a global counter instead of the id field. Yet another method relies on the fact that input/output source ports assigned to a client computer by a NAT router are very similar to those the client computer would assign to its outgoing packets. For example, Mandrake 9.2 assigns

source port 32,885 to an outbound packet, which is changed to 32,774 by a NAT router. This progression is very similar to that of the client itself: it simply involves counting up from a relatively high port number as each new source port from a client is used. Examining the general numeric range of source ports may give some insight into the type of communicating device, and a sudden change in source port numbering may indicate a new device has been plugged into the port. Determining whether the device is a router would require a large sampling of NAT router behavior, which is beyond the scope of this paper. In any case, the method is unsuccessful unless a specific source port assignment scheme can be derived from the NAT router and the scheme is shown to be different from the scheme used by the client to assign source ports to outgoing packets.

The IP id and TTL counting techniques are somewhat unreliable when used individually. Our combined method uses simple TTL counting and IP id counting and grouping. In addition, it observes the treatment of outbound packets by the operating system, allowing for more accurate detection and client counting.

3. NAT Router Detection

Examining the TTL values of packets is one of the easiest and most reliable ways to detect NAT routers. Most systems have a default TTL value on outbound packets of 64 (Linux and Mac OS X) or 128 (Microsoft Windows). The TTL is decremented by one when packets pass through a NAT router before continuing to a piece of switching equipment. The presence of a TTL value other than 64 or 128 on a switch or router serving clients on an "edge" network segment would, therefore, indicate the presence of a NAT router—or at least some network device—between the client and the switch [1]. This method can be defeated by changing the default TTL of the client to 129 or 65 using `iptables` or a similar utility. However, most users are not aware of the specifics of IP packets, so the detection method should be quite effective.

IP id counting can also be used to detect NAT routers. If packets arriving at a network port exhibit multiple, distinct IP id sequences and each of these sequences continues across multiple source ports without interruption, a NAT router serving two or more clients is present at that network port. This point is discussed in more detail in the next section on counting clients.

4. Client Counting

Most operating systems use the id field in the IP header as a simple counter for outgoing packets. The number used to start the count is arbitrary: it can be below 100 or in excess of 50,000. Windows and Mac OS X use a global packet counting scheme, in which the counter is usually incremented by one (monotonically increasing) for each outgoing packet, regardless of the protocol, stream or source port. Some unexplained gaps in IP id counting have been observed on Windows and Macintosh machines, but the gaps are often exactly 55 (never more), which can be incorporated into a counting algorithm. The IP id field is typically left untouched by the NAT router when it translates the packet. As described in [1], assigning packets with sequential id fields to "groups" of similar id fields from past packets permits the estimation of the number of clients behind a NAT.

Linux systems count packets on a per-source-port basis, i.e., a new counter is started whenever a new source port is used for an outgoing communication. This presents a problem for the IP id counting method: a single Linux machine loading five simultaneous HTTP requests results in the use of five distinct source ports, and exhibits the same traffic pattern as five Macintosh machines, each loading one of the five HTTP requests. This results in the use of one source port per client, and there is no way to definitively determine which topology reflects reality.

Combining TTL analysis (128 for Windows, 64 for Mac/Linux, 127 for Windows behind a NAT and 63 for Mac/Linux behind a NAT) and IP id analysis provides a clearer picture of the number and types of clients served by a NAT router. If there is only one IP id sequence for all outgoing traffic, then there is only a single non-Linux computer at the network port. If there are multiple id sequences that cross source-port boundaries with minimal or no interruption, there are multiple non-Linux machines behind the NAT router.

Further information can be gathered based on the types of TTLs. A mix of TTL values of 63 and 127 indicates a mixed client pool that includes at least one Macintosh or Linux machine and at least one Windows machine. While Macintosh and Linux machines both have default TTLs of 64, they can be differentiated by their IP id behavior. Similarly, while Windows and Macintosh machines have similar IP id counting behavior, they can be differentiated by their default TTLs. The IP id counting method outlined in [1] works well in identifying the number of NAT clients only if there are no Linux machines behind the NAT router. In essence, the presence of a Linux machine reduces the reliability of passive client counting.

Other factors, besides the presence of Linux machines, can create problems for the IP id counting method. Some natively little-endian systems do not put the IP id field into network-byte order (e.g., in optional IP fields), but instead leave the id in little-endian: this causes the bytes to be read backwards by the sniffer, producing strange, non-sequential values. Fortunately, this can be easily managed by implementing a simple post-processing algorithm.

The IP id method may not work because not all operating systems use the IP id field. Even if one assumes that all the clients behind a NAT router are Macintosh or Windows machines and all are counting on a global packet basis, there are still several cases in which IP id method works poorly. For example, if there is a large amount of intra-LAN or subnet traffic, there are large gaps in the IP id sequences because the intranet traffic does not reach the post-NAT sniffer. This can result in an incorrect (higher) count of the number of clients. Furthermore, if IP id values from two or more hosts are similar and near in time, it is difficult to determine which packets belong to which client [1]. This problem may be mitigated by careful real-time tuning of time tolerance and sequence tolerance. Time tolerance is the maximum time allowed between two similarly numbered packets belonging to the same sequence. Sequence tolerance is the maximum distance between IP id numbers from the last packet received that is considered to be part of a sequence.

5. Application-Level Techniques

In addition to observing the relevant IP header fields of packets, examining application-level packet patterns can help detect NAT routers and count their clients. For example, a computer connected to an always-on Internet connection often has its email client automatically check for new mail every so often. If packet sniffing reveals bursts of Post Office Protocol (POP) traffic at 4, 6, 11, 13, 14 and 16 minutes, one could surmise that two email clients, most likely on two separate computers, are automatically checking for email at 5 minute intervals: one client at 4, 9 and 14 minutes and the other at 6, 11 and 16 minutes (see Figure 2). This hypothesis can be reinforced by checking the user names submitted to the POP server in each burst within POP "Request: USER ⟨username⟩" packets. If there are two significantly different user names at the predicted times, there is a high probability that the traffic is caused by two separate computers behind a NAT router. However, if encryption is used (e.g., SSL or IPSec), such packet sniffing will probably not yield the desired information.

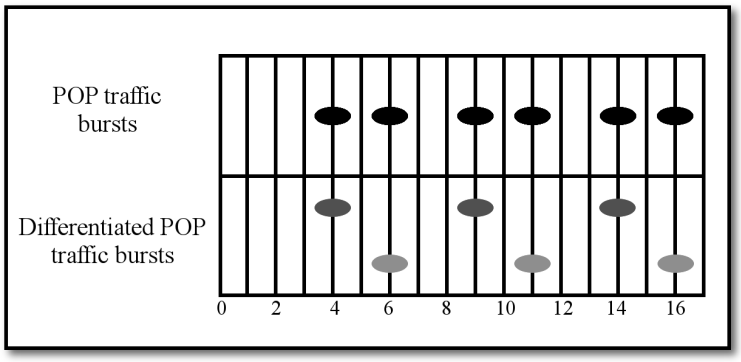

Figure 2. POP traffic bursts viewed as composed and differentiated by user name.

Note that this concept can be applied to many network activities that are likely to originate from a single computer on a periodic basis. This includes time server synchronization, operating system update checks (e.g., Windows Update) and application update checks (e.g., Norton Utilities' Live Update).

6. Conclusions

The best way to detect NAT routers is to look for numerically or logically odd TTLs on packets leaving the network port in question. The small number of packets and limited correlation required for this method to succeed makes it especially powerful. While this method is defeated by manipulating the default TTL values on client machines, it should still be able to detect the vast majority of rogue NAT routers.

IP id counting is effective at detecting NAT routers and counting the number of clients. However, research has shown that this technique occasionally works only because of the inconsistent manner in which different operating systems utilize the id field. When a per-source-port id counting machine is introduced behind a NAT router, the accuracy of the IP id method for counting hosts—or even detecting NAT routers—is greatly reduced.

Examining altered source ports is one of the least effective methods for detecting NAT routers because the altered ports are often similar to the source ports used initially by a client. Whenever a new source port is used by a client, the NAT router typically increments the most recent altered source port by one to use as the next altered source port. This monotonically increasing source port behavior is identical to that of most client operating systems. A large, sudden jump in source port

numbers not resulting from a numbering wrap-around indicates either a machine reset or a new network device connected to that port.

It is important to note that the detection and counting methods discussed in this paper assume that the NAT routers and operating systems tested are representative of the general population of NAT routers and operating systems. This assumption must be reviewed periodically to ensure its validity and the validity of methods that rely upon it. Furthermore, the methods themselves are not robust and should be audited regularly using supplementary techniques, such as physically checking all network ports.

References

[1] S. Bellovin, A technique for counting NATed hosts, *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, pp. 112-208, 2002.

[2] B. Crow, I. Widjaja, J. Kim and P. Sakai, IEEE 802.11 wireless local area networks, *IEEE Communications Magazine*, vol. 35(9), pp. 116-126, September 1997.

[3] K. Egevang and P. Francis, The IP Network Address Translator (NAT), *RFC 1631*, May 1994.

[4] A. Etter, A guide to war-driving and detecting war-drivers, SANS Infosec Reading Room, SANS Institute, Bethesda, Maryland (www.sans.org/rr/whitepapers/wireless/174.php), 2002.

[5] R. Farrow, System fingerprinting with nmap, *Network Magazine* (www.itarchitect.com/article/NMG20001102S0005), November 2000.

[6] Fyodor, Remote OS detection via TCP/IP stack fingerprinting, *Phrack Magazine*, vol. 8(54), December 1998.

[7] J. Park and D. Dicoi, WLAN security: Current and future, *IEEE Internet Computing*, vol. 7(5), pp. 60-65, 2003.

[8] L. Phifer, The trouble with NAT, *Internet Protocol Journal*, vol. 3(4), pp. 2-13, 2000.

[9] S. Russell, Detecting and locating rogue access points (www.ee.iastate.edu/∼russell/cpre537.s06/Report-Example.pdf), 2003.

[10] W. Stallings, *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2*, Addison-Wesley, Reading, Massachusetts, 1998.