Algorithms and Techniques Used for Auto-discovery of Network Topology, Assets and Services

CS4983 Senior Technical Report

Brian Chown
0254624

Faculty of Computer Science
University of New Brunswick
Canada

March 16, 2006

CONTENTS

Contents 1

Executive Summary 2

1. Introduction 3

2. Background and Motivation 3

3. Active Profiling Techniques 4
- 3.1 Network Topology Discovery 5
- 3.2 Host Link Layer Discovery 5
- 3.3 Host Internetwork Layer Discovery 8
- 3.4 Host Transport Layer Discovery 8
- 3.5 Host Application Layer Discovery 19

4. Passive Profiling Techniques 22
- 4.1 Host Link Layer Discovery 22
- 4.2 Host Internetwork Layer Discovery 23
- 4.3 Host Transport Layer Discovery 24
- 4.4 Host Application Layer Discovery 25

5. Advantages, Disadvantages, and Applicability 25
- 5.1 Active vs. Passive Techniques 25
- 5.2 Active Techniques 26
  - 5.2.1 Network Topology 27
  - 5.2.2 Host Link Layer 27
  - 5.2.3 Host Internetwork Layer 28
  - 5.2.4 Host Transport Layer 28
  - 5.2.5 Host Application Layer 30
- 5.3 Passive Techniques 31
  - 5.3.1 Host Link Layer 31
  - 5.3.2 Host Internetwork Layer 32
  - 5.3.3 Host Transport Layer 32
  - 5.3.4 Host Application Layer 33

6. Conclusions and Recommendations 34

References 36

Executive Summary

In the new generation of Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS), intimate knowledge of the underlying network is vital. Acquiring this knowledge can be accomplished through a myriad of techniques, which are categorized under two main methodologies, active and passive. Within these two categories, IDS and IPS developers have a wide range of techniques to choose from when deciding how their product will acquire the necessary network information. Through careful examination of some of the most commonly used techniques, the reader obtains a better understanding of their advantages, disadvantages and applicability. Armed with this understanding, the reader is better equipped to analyse the differences between the various IDS and IPS implementations on the market.

The active and passive methodologies mentioned have their own unique advantages and disadvantages. After having examined the various techniques, it became apparent that the active and passive techniques complimented each other perfectly. This combination of techniques could provide complete and current knowledge of the network's topology, host availability, available protocols and services, and detailed application information, with minimal impact on the network and its devices. This sort of information is not only valuable for use in an IDS or IPS, but can also be used by a network manager for other security related tasks.

1. Introduction

This report presents some of the most common and widely used techniques for attaining knowledge of networks through both passive and active methods. Along with presenting how the various techniques function, their advantages and disadvantages are compared and contrasted to provide some insight into the applicability of the various techniques to real world networks.

In section 2, the background and motivation for this report is discussed, including the relevance of these methods to modern Intrusion Detection Systems and Intrusion Prevention Systems. The next section, section 3, covers various active techniques and is presented in a format that is based on the internet protocol suite. Within section 4 is presented, in a similar format to that of section 3, the passive techniques used for attaining knowledge about networks. The fifth section of the report discusses the advantages, disadvantages and applicability of the various techniques presented in sections 3 and 4. This section is divided into three parts, the first of which compares the general differences between active and passive techniques. The second part of section 5 compares and contrasts the various active techniques with each other, and the third part looks into the strengths and weaknesses of the various passive techniques. Section 6 contains the conclusions reached during the writing of this report and the recommendations of the author.

2. Background and Motivation

Knowledge of the underlying network is becoming one of the key requirements of the new generation of Intrusion Detection Systems (IDS) and Intrusion Prevention Systems

(IPS). Lacking this knowledge can result in numerous ambiguities when interpreting alerts and making decisions on adequate responses.[1] Acquiring this knowledge can be accomplished by a variety of techniques that can be placed in two general categories: active and passive. The general methodologies behind these two categories, as well as the various techniques within each, have their own distinct advantages and disadvantages, which determine their applicability within real world implementations. Through careful examination of the characteristics of these techniques, the design of new IDS and IPS implementations can be improved, as well as making it easier for organizations to determine the best solution for their specific network security needs.

3. Active Profiling Techniques

The underlying methodology behind active profiling techniques involves actively probing a target device, for which there is a wide variety of techniques, and then analysing the device's response. Through the use of these techniques, various pieces of information about the network and its devices can be discovered, including the network topology, device availability, the protocols in use, and much more.

In this section, and throughout the report, the general format of the various sections follows that of the internet protocol suite. The internet protocol suite stack is comprised of four layers: Link, Network, Transport and Application. As we move from the Link layer to Application layer, we build upon the previous layer. In the Link layer it is determined if a connection exists between the source and destination hosts. The network layer is replaced in the report by the "internetwork" layer, which falls in between the network and transport layers. This layer is interested in the protocols that get data from

the source to the destination. The third layer is the Transport layer, where the focus is connecting applications using ports. The final layer is the Application layer, which is concerned with the actual programs and program data.

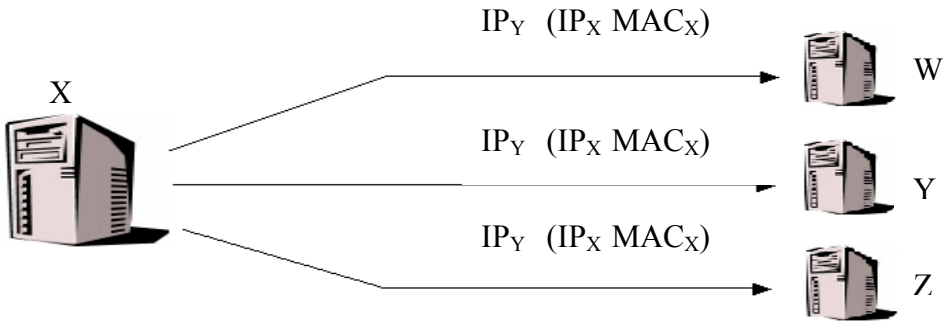
3.1 Network Topology Discovery

One common technique for discovering the topology of a network involves sending a series of groups of UDP or ICMP Echo Request packets to a destination host. The initial packets sent, have their IP header Time-to-Live (TTL) fields set to the value one and each subsequent group of packets have their TTL fields set one higher than the previous group. Each time a packet arrives at a router, the TTL field will be decremented by one and then sent on to the next router or gateway between its current location and the destination host. If a packet arrives at a router and the TTL field has the value zero, the packet is dropped and an ICMP Time Exceeded in Transit error message is sent back to the packet's source IP address. When the source host receives this error message, it sends out the next group of packets with the incremented TTL field. By examining the source IP address of the ICMP error message, the router that sent it is identified. This process continues until an ICMP Port Unreachable error message or ICMP Echo Reply, depending upon which type of packet is used, is received from the initial destination host. This produces a map of all the routers, gateways and hosts between the originating host and the destination host. Continuing this process for other hosts on the network, a complete map of all active hosts can be obtained. For this technique to function correctly, it requires that there be no prohibitive filtering or packet loss on the network.[2]

3.2 Host Link Layer Discovery

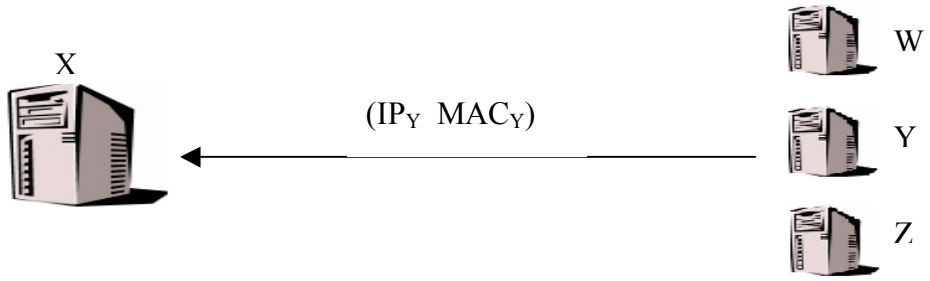
ARP Querying

The Address Resolution Protocol (ARP) was developed to facilitate the linking of IP addresses to physical addresses, also known as MAC addresses, on a network. When a host wishes to communicate with another host on the network but only has knowledge of the other host's IP address, it must first learn the MAC address of the host it wishes to communicate with. As shown in Figure 1a, host X accomplishes this by sending out an ARP request broadcast to every host on the network. The ARP request contains the IP address of host Y, the host for which the MAC address is being sought, and the IP and MAC addresses of host X so that host Y will know which host to send the reply to. In



ARP Broadcast by Host X
Figure 1a.

Figure 1b, host Y has seen its IP address in the ARP broadcast and responds to host X with its IP and MAC addresses. All the other hosts on the network who received the ARP broadcast simply ignore it since the IP address in the broadcast did not match their own.[3]



ARP Reply from Host Y
Figure 1b.

Through systematic querying of all the IP addresses on a network, a database containing the IP addresses and associated MAC addresses for all active hosts can be built. However, it is important to note that this technique relies on the network having a broadcast method, which is not something implemented by all LANs.[3]

ICMP Echo Scan

This technique involves sending an ICMP Echo (ICMP type 8) datagram to the destination host and waiting to see if there is a response. If the target host is active, an ICMP Echo reply (ICMP type 0) is sent back from the target host. If the source host receives no response to its ICMP Echo request, either the target host is inactive or the ICMP protocol is being filtered.[4] Unfortunately, the ability to differentiate between the two causes of a failure by the target to respond with an ICMP Echo reply, is beyond the ability of this scanning technique. It is also possible that an ICMP error message will be returned, such as a: ICMP Host Unreachable or ICMP Destination Unreachable port unreachable error. In the case of an ICMP Host Unreachable error, it indicates that the targeted host is either temporarily down or does not exist. For the port unreachable error, it can be determined that the host is alive and reachable, but the port is closed.[5] One of the disadvantages of this scanning technique is that in the case where no response is

received from the target host, it cannot be determined if the lack of response was due to an inactive or non-existent host or that the ICMP protocol is simply being filtered. However, if access to a network is available without any filtering issues along the connection path, such as firewalls, the ICMP Echo scan technique can be a useful way of creating a list of active stations on the network.[6]

3.3 Host Internetwork Layer Discovery

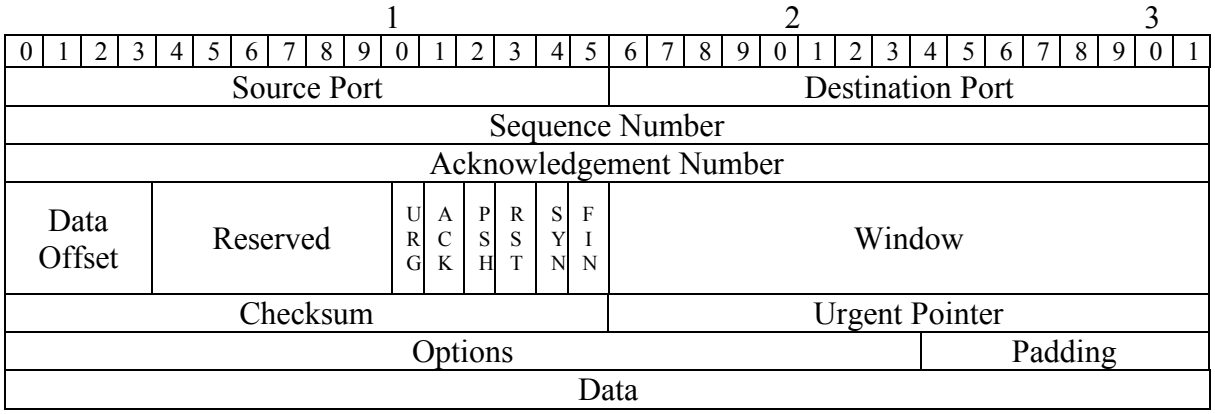
The purpose of the IP protocol scan is to identify the various IP protocols (TCP, ICMP, IGMP, etc.) that are supported by the target host. To accomplish this, a series of IP packet headers is sent to a host, each of which contains a different value for the IP protocol field, until all of the possible protocol values have been used. If the target host responds using one of the protocols, then that protocol is available on the host. If instead an ICMP protocol unreachable error message is returned, then the protocol associated with that message is not available on the host.[7]

3.4 Host Transport Layer Discovery

To understand how the various scanning techniques make use of the Transmission Control Protocol and the User Datagram Protocol it is important to understand some of the standard practices for TCP and UDP communication as set out in RFC 793 and RFC 768 respectively. Some of the relevant portions of RFC 793 and RFC 768 are outlined below.

TCP Header Format

"TCP segments are sent as internet datagrams. The Internet Protocol header carries several information fields, including the source and destination host addresses. A TCP header follows the internet header, supplying information specific to the TCP protocol."



TCP Header Format
Note that one column represents one bit position.
Figure 2.

Source Port: 16 bits
The source port number.

Destination Port: 16 bits
The destination port number.

Sequence Number: 32 bits
The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.

Acknowledgment Number: 32 bits
If the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent.

Control Bits: 6 bits (from left to right):
URG: Urgent Pointer field significant
ACK: Acknowledgment field significant
PSH: Push Function
RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: No more data from sender

Window: 16 bits
The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.[8]

TCP Functionality

"The procedures to establish connections utilize the synchronize (SYN) control flag and involve an exchange of three messages. This exchange has been termed a three-way handshake."[9] The first message sent in a typical three-way handshake between device A and device B is a TCP segment with the SYN control bit set and an initial sequence number. This segment is sent from A to B, and is often referred to simply as a "SYN". In response to the SYN, the second message is sent from B to A and it has the SYN and ACK control bits set, as well as an initial sequence number for device B. To complete the final part of the three-way handshake, device A sends a segment back to B with the ACK bit set.[10] The second message in the handshake is often referred to as a "SYN/ACK", and the third message as simply an "ACK". With the completion of the three-way handshake, a connection has now been established between devices A and B. At this point the devices can begin transmitting segments containing "data" to each other. Shown below is an example of the communication that takes place between TCP device A and TCP device B in a typical three-way handshake.

	TCP A		TCP B
1.	CLOSED		LISTEN
2.	SYN-SENT	--> <SEQ=100><CTL=SYN>	--> SYN-RECEIVED
3.	ESTABLISHED	<-- <SEQ=300><ACK=101><CTL=SYN,ACK>	<-- SYN-RECEIVED
4.	ESTABLISHED	--> <SEQ=101><ACK=301><CTL=ACK>	--> ESTABLISHED
5.	ESTABLISHED	--> <SEQ=101><ACK=301><CTL=ACK><DATA>	--> ESTABLISHED

Basic 3-Way Handshake for Connection Synchronization
Figure 3.

In line 2 of figure 3, TCP A begins by sending a SYN segment indicating that it will use sequence numbers starting with sequence number 100. In line 3, TCP B sends a SYN and acknowledges the SYN it received from TCP A. Note that the acknowledgment field indicates TCP B is now expecting to hear sequence 101, acknowledging the SYN which occupied sequence 100. At line 4, TCP A responds with an empty segment containing an ACK for TCP B's

> SYN; and in line 5, TCP A sends some data. Note that the sequence number of the segment in line 5 is the same as in line 4 because the ACK does not occupy sequence number space (if it did, we would wind up ACKing ACKs!).[11]

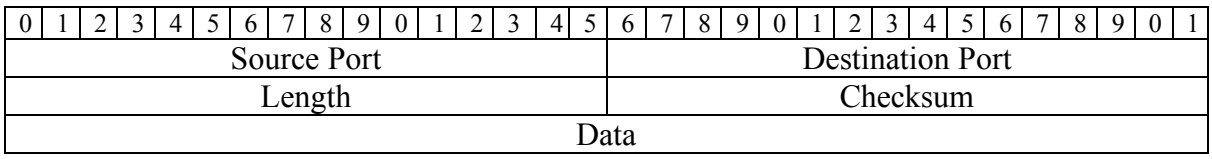
Another important area of the Transmission Control Protocol to understand is how a device responds to segments that have the reset control bit set. When a segment with its reset bit set is received, if a connection does not exist, the segment is ignored. If a connection does exist, the connection is aborted. Also of importance are some of the ways in which a device can be prompted to respond with a reset segment. "As a general rule, reset (RST) must be sent whenever a segment arrives which apparently is not intended for the current connection." There are two general situations in which resets should be sent that are exploited by various port scanning techniques:

1. If the connection does not exist… then a reset is sent in response to any incoming segment except another reset. In particular, SYNs addressed to a non-existent connection are rejected by this means.

2. If the connection is in any non-synchronized state… and the incoming segment acknowledges something not yet sent (the segment carries an unacceptable ACK),… a reset is sent.[12]

UDP Functionality

As can be seen in Figure 4, the UDP header has far fewer fields than its TCP counterpart. Within the UDP header there is a field for the source port and the destination port, which have meaning in the context of the IP addresses contained within the IP header that accompanies the UDP datagram. The remaining fields contained in the UDP header are not relevant in the context of the scanning techniques presented in this paper.

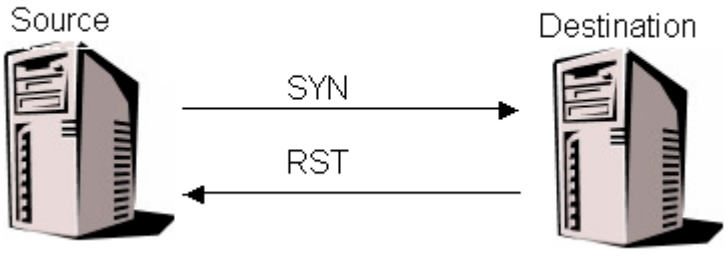


User Datagram Header Format[13]
Figure 4.

Unlike TCP, UDP is not a connection-based protocol but rather a transaction oriented one. What this means is that no "handshaking" like that of TCP takes place between two hosts communicating using UDP. When a host wants to send a UDP datagram to another host, the datagram simply needs to be formed and sent. The destination host requires no advance notice. If there is no UDP service listening on the targeted port, an ICMP Port Unreachable error is the expected response from the destination host.[14]

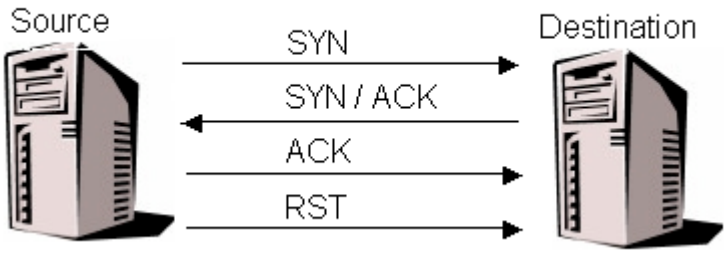
TCP connect() Scan

Generally viewed as the most basic scanning technique that uses TCP, connect() scanning makes use of the operating system's native method of establishing connections to remote devices. This technique involves completing the full "three way handshake" described by TCP, the same process undertaken by other network devices that rely upon TCP for communication across the network. A scan of this type sends a SYN packet from the source to a port on the destination host. As shown in Figure 5a, if the port is closed the source expects a RST packet as a response from the destination host and the scan of that port is complete. However if the port is open, as seen in Figure 5b, the source expects



TCP connect() Scan on Closed Port
Figure 5a.

to receive an ACK frame and a SYN frame from the destination host. The source host will then send an ACK frame of its own to the destination host, completing the connection and immediately following up by sending a RST packet to the destination host to terminate the connection.[6]

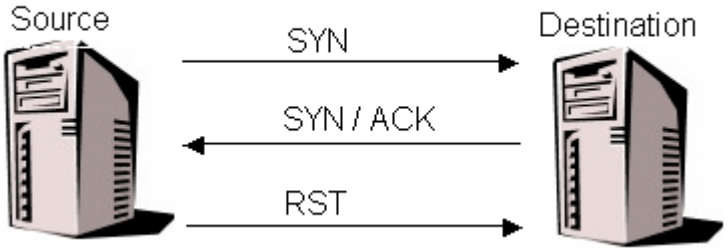


TCP connect() Scan on Open Port
Figure 5b.

TCP SYN Scan

The SYN scanning technique, often referred to as "half open" scanning, is similar to the TCP connect() scan except that a complete connection is never established to an open port. A SYN scan begins by sending a SYN packet to a port on the destination host, prompting the destination host to respond with a RST packet if that port is closed, just like the communication shown above in Figure 5a for a TCP connect() Scan on a closed port. If the port is open, the communication between the two hosts will look like that of Figure 6. The destination host will respond to the source's SYN packet with an ACK frame and SYN frame. In response to the ACK frame, the source host will immediately send a RST packet, thereby preventing a connection from being completed with the

destination host. Due to the fact that no connection is completed, this type of scan is



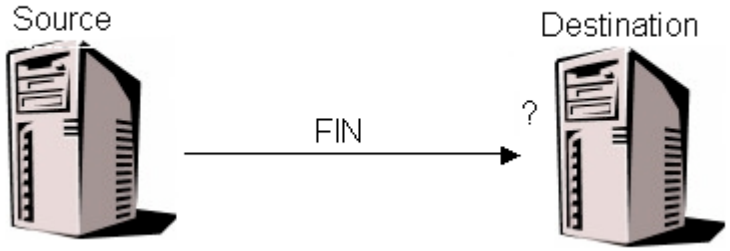
TCP SYN Scan on Open Port
Figure 6.

more likely to go unnoticed by the destination system.[15] As a result of this technique's deviation from the normal TCP protocol, the need for the ability to create custom SYN packets is introduced. While the TCP connect() scan is generally viewed as the most basic scan, the "TCP SYN scan is the most common scan to use because it works on all networks, across all operating systems, and it's invisible to applications."[6]

Control Bit Scans

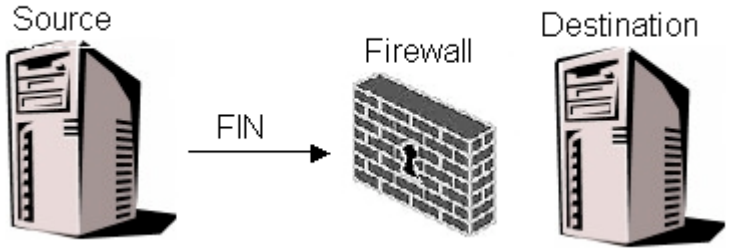
Scans of this type involve the manipulation of the six control bits (URG, ACK, PSH, RST, SYN, FIN) that are present in the TCP header of a packet sent to the target port on the destination host. In an attempt to confuse the destination host, the control bits are set in combinations that should never occur, or that represent situations that should never occur in the real world. Three frequently used scans of this type are the FIN scan (only the FIN bit is set), the Xmas scan (FIN, URG, and PSH bits are set), and the Null scan (none of the bits are set). With none of the usual TCP handshake process having taken place previously, one of these packets with a mangled TCP header is sent to a port on the destination host. Figure 7a shows a FIN scan on an open port where the destination host receives a packet with the FIN bit set, which indicates that no further transmissions are

coming. The destination host, having no existing connection to associate the FIN packet with, sees the packet as an anomaly and simply drops the packet. While the lack of a



FIN Scan on Open Port
Figure 7a.

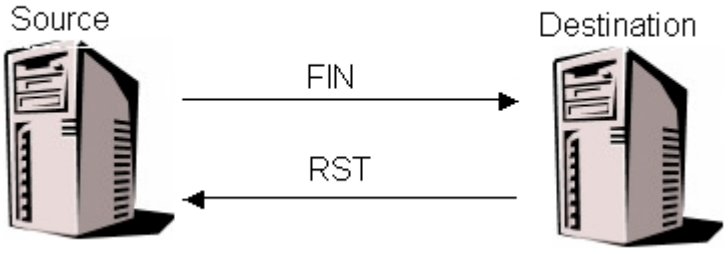
response from the target host is indicative of an open port, it is also possible that the FIN packet was simply filtered by a firewall causing the packet to be dropped before reaching its destination as seen in Figure 7b. As a result, this type of scan cannot differentiate



FIN Scan on Filtered Host
Figure 7b.

between an open port and a dropped packet due to filtering between the two devices. In Figure 7c a FIN scan is run on a closed port and provided the target host's TCP/IP stack conforms to the RFC 793 Transmission Control Protocol, a RST packet is sent back to the source host in response to the FIN packet. Adherence to the RFC 793 protocol is an

important caveat for this type of scan because there are various systems that do not



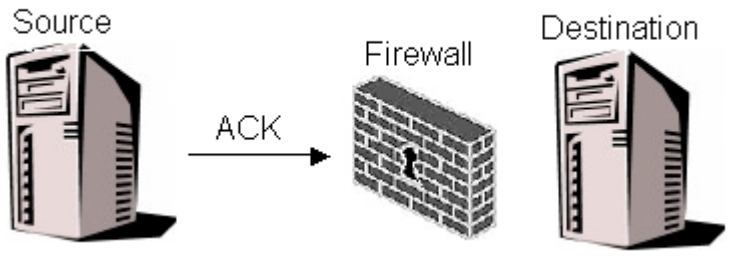
FIN Scan on Closed Port
Figure 7c.

conform to the protocol, notably Windows-based systems which will respond to this type of scan with a RST packet whether the targeted port is open or closed. As a direct result of this, it is possible to make some assumptions based on the results of this type of scan. For example, if an open|filtered port is discovered, it can be assumed that the target is not a Windows-based system. On the other hand, if when scanning multiple ports on a host, they are all showing up as closed, it could be that the target host is a Windows-based system and therefore another type of scan will be required to determine the true port status on the host. Another possible deduction is when a scan across multiple ports on the host is showing all the ports as open|filtered, then there is a good chance that the scan has encountered a firewall and the packets are all being dropped at the firewall.[6]

ACK Scan

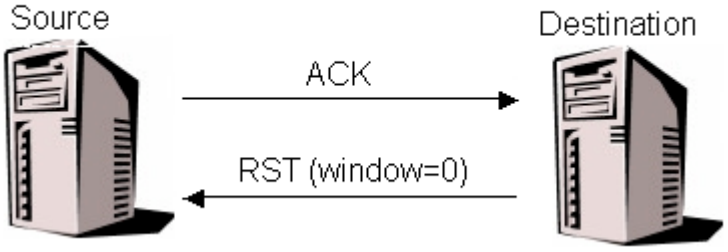
The ACK scan is similar to other scan types that involve manipulation of the control bits of the TCP header. However, in the case of the ACK scan, the values within the header are similar to those seen within typical network traffic. This scan type is very useful for discovering the filtered status of hosts and can accomplish the task while creating a minimal amount of network traffic because no connection to the destination host is ever

created. Inside the TCP header of a packet, the ACK control bit is set, as well as random acknowledgement and random sequence numbers are assigned to their respective fields. The ACK scan then sends a single packet to the target port on the destination host and then examines the response, or lack thereof, from the destination host. If the targeted port responds with a RST then the port is considered to be unfiltered. However, if there is no response from the target port or an ICMP destination unreachable message is returned, then the port is categorized as filtered, as seen in Figure 8a.[6]

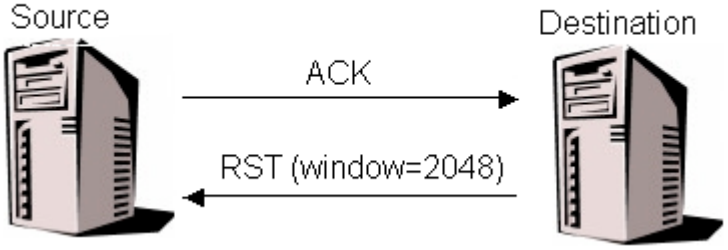


ACK Scan on Filtered Port
Figure 8a

Along with determining the filtered or unfiltered state of the target, it is also possible to determine whether there is a TCP service listening on the targeted port, by examining the "window" field within the TCP header of the RST packet returned from an unfiltered host. When a closed port responds to the ACK with a RST, the window field has a value of zero (Figure 8b), but in the case where the port is open, some operating systems will respond with a RST that has the window field set to a non-zero value (Figure 8c). This lack of consistency allows open ports to be identified, however awareness of this inconsistency has increased and many previously vulnerable operating systems have



ACK Scan on Closed Port
Figure 8b.



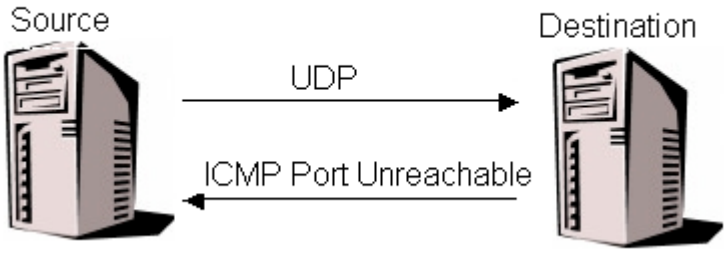
ACK Scan on Open Port
Figure 8c.

developed patches that remove the inconsistency. Even so, in cases where an open port can be found, that inconsistency can be used to narrow down the number of possible operating systems present on the host.[6]

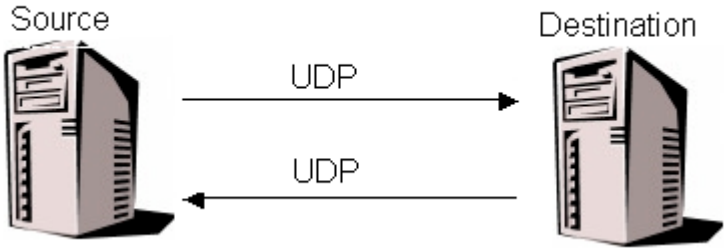
UDP Scan

As discussed previously, the UDP protocol is more simplistic than that of TCP, requiring none of the complicated "handshaking" involved in TCP communications. A "UDP scan works by sending an empty (no data) UDP header to every targeted port. If an ICMP port unreachable error (type 3, code 3) is returned, the port is closed. Other ICMP unreachable errors (type 3, codes 1, 2, 9, 10, or 13) mark the port as filtered. Occasionally, a service will respond with a UDP packet, proving that it is open. If no

response is received after retransmissions, the port is classified as open|filtered. This means that the port could be open, or perhaps packet filters are blocking the communication."[16] Even though the UDP scan does not have the overhead involved with some of the other scans, the time required



UDP Scan on Closed Port
Figure 9a.



UDP Scan on Open Port
Figure 9b.

for a UDP scan to complete can be significant due to the rate limiting of ICMP port unreachable messages that many operating systems implement. As an example, some Linux systems set a limit of one packet per second, which translates into more than eighteen hours for a full 65,536 port scan on a single host.[16]

3.5 Host Application Layer Discovery

The basic methodology behind the fingerprinting technique involves probing a target host with a variety of scanning techniques, prompting the host to respond. The responses are then analysed and compared to responses within a database of response "fingerprints" from known applications. To implement this technique, the "fingerprint" database must first be built or acquired, as well as continually updated if this technique is to be effective and accurate. Along with the "fingerprint" database, a database of "probes" to coerce data from the host must also be developed and maintained. These probes define the scan types and the specific characteristics of the packets that are to be sent to the target host. The fingerprinting technique can be used to identify the applications available on a host as well as the operating system (OS), however the particular probes involved in identifying them vary.[6]

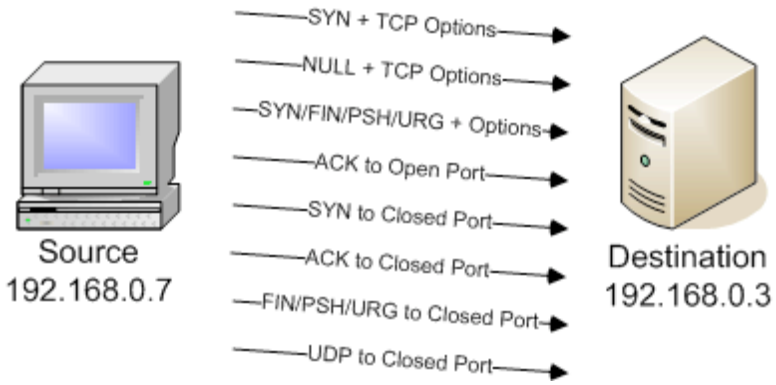
Identifying the applications available on a host, first requires that the ports they are available on be identified. This can be accomplished through various combinations of the scanning techniques discussed previously. Once the open ports have been located, probing of the services and establishment of sessions with the application can begin. This probing is quite invasive, however it is necessary to obtain the information from the application so that there is something to match up with the fingerprint database. Provided the fingerprint database and the probes are well developed, a significant amount of information can be learned about the applications on the host. This information can then be used to aid in things such as patch management, location of non-compliant software, and verifying an organization's licensing agreements.[6] Some examples of the information that can be discovered by this technique can be seen in Figure 10.

```
PORT     STATE  SERVICE       VERSION
80/tcp   open   http          Apache
httpd 2.0.53 ((Win32))
135/tcp  open   msrpc         Microsoft Windows
msrpc
139/tcp  open   netbios-ssn
445/tcp  open   microsoft-ds  Microsoft Windows XP
microsoft-ds
520/tcp  open   efs?
3306/tcp open   mysql?
3389/tcp open   microsoft-rdp Microsoft Terminal
Service
```

Excerpt From Scan Results Using nmap[6]
Figure 10.

The fingerprinting process for determining the OS of a host is much less invasive than that of determining information on the available applications. Where discovering the application details involves opening sessions with the target host, OS fingerprinting can be accomplished without opening a single session. A variety of methods can be used to acquire the pieces of information that when put together can be compared against the known fingerprints database. Some of the common methods involve manipulating TCP segments in various ways, much like those used to discover open ports, except that the responses from the target device are examined for particular field values, option combinations in the header, etc… that will help narrow down the possible operating system details on the target device. Another common method that can be used involves sending a UDP frame to a closed port and then examining various attributes of the ICMP port unreachable reply that is sent back by the target device. Through the use of various

combinations of these methods, some of the information that can be determined from OS fingerprinting is: the device type, the OS running on the device, OS patch information, TCP sequence predictabilities, and IPID sequence generation. The attainability and accuracy of the information depends on various factors such as the particular probes used, the susceptibility of the target device to those probes, and whether there are both open and closed ports on the target device.[6] Figure 11 shows a subset of the possible methods used for probing a host during OS fingerprinting. The particular methods for OS



Partial OS Fingerprinting Process Used by nmap[6]
Figure 11.

fingerprinting that are implemented in the various available scanning software vary from one to the other, however many of the techniques are focused on examining differences in TCP/IP stack implementations between operating systems.

4. Passive Network Discovery

The general methodology behind passive discovery techniques involves placing a sensor somewhere on the network so that it monitor network traffic as it flows past the sensor. By examining the type and behaviour of the traffic, it is possible to determine various sorts of information about the devices on the network. Unlike its active counterpart, passive techniques do not communicate with the devices on the network.

4.1 Host Link Layer Discovery

The purpose of host discovery is to locate the active devices on a network. As a device produces any kind of traffic on a network, that traffic can be detected and used to identify the device as an active element on the network. One protocol in particular that is used to discover IPv4 hosts on a Local Area Network (LAN), is the Address Resolution Protocol, or ARP for short. The purpose of the ARP protocol is to map IP addresses to hardware addresses. When a host is looking for the hardware address that goes with a particular IP address, it sends out an ARP request across the network and waits for the host with that IP address to send back its hardware address. The broadcasting of these ARP requests allows a sensor anywhere on the network to analyse the packets and compile a list of active IP addresses and their associated hardware addresses. Through the use of constant passive network monitoring, resource inventory and awareness of new additions to the network can be maintained.[1]

4.2 Host Internetwork Layer Discovery

Knowledge of the protocols being used by a host can aid in monitoring network activities, as well as helping to identify the role of a particular host on the network. The passive technique for protocol detection makes the assumption that if a sender uses a particular protocol then it follows that the sender supports that protocol. While this assumption is generally true, false positives can occur in cases where packet crafting is involved, such as a host running a port scanning tool which implements some of the active techniques discussed previously. Fortunately this situation is usually easily detectable since it is uncommon for a host to support such a wide range of protocols over IP. Some of the protocols of interest are carried on top of the data link layer (IPv4, IPv6,

IPX, CDP, STP, AppleTalk, Netware, NetBIOS, etc.) while others on top of IP (IGMP, OSPF, TCP, UDP, ICMP, etc.). For the protocols on top of IP, the IP header is examined for the value in the "Protocol" field for IPv4, or the "Next Header" field for IPv6. Within the field is stored a protocol number that identifies the specific protocol being used for that packet. The other set of protocols, those on top of the data link layer, are similarly identified by the value of specific fields, however the fields of interest vary with the frame format.[1]

4.3 Host Transport Layer Discovery

Passive discovery of open TCP ports involves monitoring the set-up of the connection between the two devices. If a SYN/ACK is sent in response to a SYN, the TCP header of the SYN/ACK is examined and the port indicated by the source port field is assumed to be open, though this assumption is not always accurate. For example, in the case of a FTP file transfer there are two communication channels, one for data and one for communication. In the case of the data channel, when it is created the source port will only be open until the connection terminates. In this case and in others, the source port of the SYN/ACK packet is not associated with a network service, however it is generally easy to implement methods to differentiate between the two situations, especially if the particular network monitoring tool employs mechanisms for matching packets that belong to the same connections.

In the case of UDP ports a different method must be employed due to the lack of synchronization involved in UDP communication. Any UDP port less than 1024 that is transmitting packets can simply be assumed to be open. While this leaves the status of

UDP ports operating on higher port numbers unknown, if any of them are sending or receiving communications using UDP then they can be marked as possibly hosting network services and verified through other means.[1]

4.4 Host Application Layer Discovery

Unlike the active fingerprinting technique, passive fingerprinting does not send a single packet to the target device. Through the use of a network monitoring tool, packets from normal network traffic can be viewed and then analysed for characteristics that can then be compared to pre-existing fingerprints, without any attempt to force the target device into sending any packets.[17] Many of the characteristics and behaviors that are watched for among the network traffic are the same as those used in active fingerprinting, such as the values of various TCP header fields. Some further examples of inconsistencies that can be used for passive fingerprinting are: the delay between retransmission attempts for unanswered packets, the number of retransmission attempts, and the default value of the "Target Hardware address" field within ARP requests.[1] This wide range of possible inconsistencies among different applications, from how they respond to certain packet types[18] to all the little details that go into making up the various protocol headers in the packets they transmit, can often allow various applications to be identified very accurately, even down to specific versions.

5. Advantages, Disadvantages, and Applicability

5.1 Active vs. Passive Techniques

The most obvious difference when comparing active vs. passive techniques is their effect on the network. While the monitoring methods employed in passive techniques create no

additional traffic on the network, all active techniques create varying amounts of additional network traffic. As the number of hosts and ports to be scanned increases, the amount of increased traffic can interfere with normal network and host operation. A scan of all 65,536 ports/host on a class B network using the TCP SYN scan, which creates less traffic than some of the other techniques, can introduce upwards of 170 gigabytes of traffic.[20] Since passive techniques operate using only existing network traffic, they do not suffer from the same sort of difficulties in this regard as do their active counterparts. This lack of disruption to the network and devices connected to the network is one of the main advantages of passive techniques.

Another major difference between active and passive techniques is in the acquisition of information about the network. Active techniques can be initiated at any time to provide a complete picture of the host or network in question, only requiring the time needed to run the scan. Unfortunately, the information obtained in this manner is only a snapshot of the network and can become outdated shortly thereafter. What this means is that any changes to the network that take place after a scan is run will not be accounted for until another scan is initiated. Contrast this with passive techniques that allow the picture of the network to be updated continuously. However it is important to note that while the picture of the network may update continuously, it does not necessarily represent a complete picture of the network. Passive techniques are only able to detect a host on the network if that host is communicating on the network and the traffic from the host is passing through the point on the network that the sensor is monitoring. As a result of

these limitations, it generally takes longer to profile assets using passive techniques than it does using active ones.

5.2 Active Techniques

5.2.1 Network Topology

The Time-to-Live technique used for network topology discovery is an effective and commonly used technique, however there are a couple important factors to consider when using it. The first of these factors is that as the size of the network increases, so does the time required to map the network, and for larger networks it can become prohibitive. The second important consideration is how reliably the packets can be delivered across the network. If there is prohibitive filtering employed on the network or excessive packet loss, then the accuracy of the mapping will be compromised, if not completely ineffective. Even with these limitations, this technique remains a viable solution for mapping many real world networks.

5.2.2 Host Link Layer

Creating an inventory of active stations on a network can be accomplished using either of the two techniques discussed in Host Link Layer Discovery, however both have disadvantages that make them ineffectual in certain situations. The first technique discussed, ARP querying, requires that the network have some method of broadcasting an ARP request to every host on the network. Unfortunately not all LAN technologies implement a broadcast option, which makes this technique ineffectual on a non-broadcast LAN. The ICMP Echo scan on the other hand, does not suffer from this same

requirement and would be a possible solution when dealing with a non-broadcast LAN. The ICMP Echo technique is widely used on real world networks, providing a fast means of determining host availability while creating a minimal amount of traffic on the network. The downside to this technique stems from this technique's inability to distinguish between a lack of response caused by an inactive host, and a lack of response as a result of a filtered communication channel between the initiator of the scan and the target host. While the speed and minimal traffic created by this technique make it an attractive choice, the ICMP protocol is susceptible to abuse and in response to this vulnerability, "ICMP is one of the most filtered protocols in enterprise networks."[6] To take advantage of this technique's positive attributes, it's vital that an unfiltered path exist between the scanning device and the other hosts on the network.

5.2.3 Host Internetwork Layer

The IP Protocol scan is an effective means of discovering all the protocols on a host, though it does not accomplish this as efficiently as it could. Of the 256 protocol codes the scan uses for each host, currently only 55% of the codes represent valid protocols.[19] Inefficiency aside, knowledge of the protocols in use on the target device can be beneficial in helping to determine if the device is a router, printer, or workstation.[6] While it is not part of the intended purpose of the IP Protocol scan, it can be deduced that a device that responds to any of the protocols is active, though this is far from the most efficient method for obtaining such knowledge.

5.2.4 Host Transport Layer

One of the major concerns with any active scanning technique is its impact on the network due to the additional traffic they create during the course of a scan. By their very

nature, active scans create some amount of traffic during their operation, however that amount can vary greatly between the various types of scanning techniques and can be a determining factor when selecting a technique for use in a particular situation. Of the techniques discussed, the Control Bit and ACK scanning techniques create the least amount of network traffic, neither scan requiring the communication overhead normally associated with the TCP protocol they make use of, and neither of the techniques involve opening any sessions with the target host. However, both techniques suffer from their limited applicability across the full range of operating systems. At the other end of the spectrum is the TCP connect() scan which involves the full TCP "handshaking" process. This not only creates the communication overhead involved with establishing a connection, but it also means that a session is opened with the target host and even though the session is terminated immediately, normal application activities are already initiated prior to termination of the connection. For example, before the terminating RST packet can be received, a login screen or introductory page may already have been supplied by the application, which causes unnecessary use of system resources.[6] The other two techniques discussed, TCP SYN and UDP scanning, fall in between the previously mentioned techniques in terms of the additional traffic they create.

Another important consideration is the applicability of the various techniques to different operating systems. As mentioned previously, the Control Bit and ACK scans both have limitations in this regard. The Control Bit scan relies on the adherence of TCP/IP stack implementations to the TCP protocol as laid out in RFC 793, this however is not the case and many operating systems do not conform to the standards set out in RFC 793, most

notably Microsoft's Windows OSes. On a Windows-based host, a Control Bit scan will categorize all ports as closed regardless of the actual status of the port. This obviously makes the technique useless on Windows-based devices. It is also important to note that some operating systems implement rate limiting on the number of RST packets sent to a single host. This can result in false positives (identifying a port as open when it is not).[6] Unlike the Control Bit scan, the ACK scan technique relies on an inconsistency in the behaviour of OSes rather than relying upon them to adhere to standards. However, the applicability of the ACK scan is diminishing as more and more operating systems are updated to remove the inconsistency. The Control Bit and ACK scans are not the only techniques that suffer from limited applicability, many operating systems now implement ICMP rate limiting which drastically increases the run time of the UDP scan. ICMP rate limiting does nothing to hinder the proper functioning of the UDP scan technique, but the increased run time it causes can be prohibitive, as shown by the example given in section 3.4 where a scan of all 65,536 ports on a single host can take over 18 hours. Linux and Solaris are a couple of the operating systems that have implemented ICMP rate limiting and therefore the UDP technique is less desirable for scans on hosts running those Oses. However, Microsoft has not implemented ICMP rate limiting in its operating systems and therefore the UDP scan remains a viable technique for use on many real world networks. While the three previously mentioned techniques all have some restrictions on their applicability, one of the biggest advantages of the TCP connect() and TCP SYN scans is their applicability to all operating systems.

5.2.5 Host Application Layer

The fingerprinting technique relies heavily upon having an extensive "fingerprint" database and a well-developed set of "probes" for obtaining the information from a target host for comparison with the known fingerprints in the database. Even after the "fingerprints" and "probes" are acquired, there is still considerable effort required to keep them up-to-date as new applications and new methods of probing them for information are discovered. While the effort involved in maintaining the support mechanisms for the fingerprinting technique is a minor disadvantage, the more significant concern with active fingerprinting rests in the invasiveness of the technique. The probing of an application on the host to acquire the necessary information to create an application "fingerprint" for comparison with the known "fingerprints" database, involves opening a session with the application to facilitate the acquisition of that information. Compared to the scan types used for discovering information about the other host layers, the fingerprinting technique creates significantly more network traffic. Despite these disadvantages, there is a wealth of valuable information that can be obtained using the fingerprinting technique. One of the common applications of the technique is for identifying the operating system running on a host, even down to the specific version number. The same can be accomplished for other applications on the host, providing a convenient means of software and patch management. The ability to remotely locate outdated and non-compliant software over a network is a valuable asset in terms of managing network security.

5.3 Passive Techniques

5.3.1 Host Link Layer

Monitoring network traffic, and in particular ARP traffic, is an effective method of locating active hosts on a network, even those hosts which have low uptime that might be

missed when using active techniques. Through continuous monitoring an inventory of network elements can be maintained in real time while having zero impact on the performance of the network. The ability to have real time information about the network with no negative impact to the network is a big advantage for passive host discovery, however there are significant disadvantages to this technique as well. If a host is not communicating on the network, or the host's communications are not passing through the monitoring point, then this technique will be unaware that the host exists and consequently will only be able to provide a partial inventory of the network elements. Another issue that arises with this technique is the lack of control over how long it takes to discover elements on the network. With no way to actively seek out information about a particular device, the monitoring system must simply wait until the device sends some traffic past the monitoring point.

5.3.2 Host Internetwork Layer

Passive detection of the protocols supported by the hosts on a network first requires the hosts to use the various protocols they support. As a result of this, until every host on the network has used each of the protocols it supports, this passive technique can only provide a partial picture of the protocols in use on the network. Even when a complete picture of all the protocols in use is obtained, there may be some hosts that do not support all the protocols they are reported to. These "false positives" are caused by the assumption this technique makes when a host is observed using a particular protocol. When traffic from a host uses a protocol, this passive protocol detection technique makes the assumption that the host must support that protocol, which is generally true but not always. While these characteristics of this passive protocol detection technique are worth

noting, this technique still benefits from continuous updates in real time and no increased network traffic. Even with the disadvantages mentioned, this technique for passive protocol discovery remains not only viable, but also desirable in many situations.

5.3.3 Host Transport Layer

For passive discovery of available TCP based services on a network to take place, a client must request the service from a host on the network and have the host grant the request. Once again this leads to a situation where the information provided by a passive technique is incomplete, at least until such time that all the services on every host on the network have sent a SYN/ACK packet in response to a client's request for service. There are also situations where the source port in the SYN/ACK is not indicative of an available service on that port, however there are usually methods by which these situations can be detected and the availability of a service on the port properly determined. In situations where the services on the network are fairly active, and a relatively complete picture of available services can be obtained in a reasonable amount of time, the lack of any impact on the network and continuous updates of this passive technique outweigh the negatives.

Passively discovering UDP ports involves making assumptions about the existence of a service based on observed behaviour and on the port number in question. This technique obviously is not as reliable as might be desired, and it also is limited to identifying UDP ports in the sub-1024 range. Possible services indicated by UDP communication on higher port numbers must be confirmed by some other means. While this passive technique benefits from no increased traffic and continuous updates, the disadvantages detract greatly from this techniques desirability. In fact this technique highlights how a

combinations of active and passive techniques could be used to provide a better solution than a purely passive approach. Using the passive technique to locate ports of interest, and then running an active scan on those particular ports, would provide better results while keeping additional network traffic to a minimum.

5.3.4 Host Application Layer

Similar to its active counterpart, the passive fingerprinting technique requires the development of a "fingerprint" database prior to using the technique. This technique also requires the development of some tests that can extract the relevant information from packets that have been captured on the network. The database and the tests must be continually updated as new tests and fingerprints are discovered. Passive fingerprinting also suffers from one of the common problems associated with passive techniques, a lack of control over the pace of discovery. If the packets containing the necessary information for comparison to the fingerprint database are not available to the monitoring system, no deductions about the applications available on the various hosts can be made. Assuming that the network's hosts are reasonably active, a relatively complete and up-to-date inventory of host applications can be maintained using this technique. Under favourable circumstances, passive fingerprinting can provide all the information that active fingerprinting can, with none of the invasive probing and increased network traffic.

6. Conclusions and Recommendations

The large number of active scanning techniques can be partly attributed to all the time people have put into developing and investigating them over the years. As new discoveries were made, old techniques were improved or replaced until we arrive at the set of techniques we have today. Even with years of refinement, active techniques still

invariably have an impact on the network. It is an unavoidable part of the underlying methodology, and as such its effect can be minimized but never full eliminated.

Passive monitoring techniques are a more recent development in the area of network information discovery. These techniques have not had as much time to mature as their active counterparts, but there are some parts of the underlying methodology that will continue to impose their limitations on future passive techniques. Passive techniques will always be dependent upon the timely availability of network traffic for analysis. Without any traffic to analyse, there is no information about the network to be gleaned from it.

Having thoroughly examined all the active and passive techniques covered in this report, it is apparent that no single technique is suited for the diversity of situations involved in network security. Even when comparing active and passive methodologies, neither category is suitable on its own in all circumstances. In the author's opinion, the obvious solution for next generation Intrusion Detection and Intrusion Prevention systems is to combine the best characteristics of the active and passive techniques. This combination of techniques could provide complete and current knowledge of the network's topology, host availability, available protocols and services, and detailed application information, with minimal impact on the network and its devices. This knowledge could provide the solid understanding of what constitutes normal network traffic and behaviour, which is critical for IDS and IPS systems to accurately and efficiently detect possible threats and respond appropriately.

References

[1] Annie De Montigny-Leboeuf and Frédéric Massicotte. *Passive Network Discovery for Real Time Situation Awareness.* Communication Research Centre Canada, Ottawa, Ontario, 2004.

[2] Ofir Arkin. *ICMP Usage in Scanning: The Complete Know-How,* Version3.0. Sys-Security Group, June 2001, p 75. <http://www.sys-security.com/index.php?page=icmp>

[3] Simon Oosthoek. *Survey of Multicast Support for IP over ATM for Implementation Purposes*. July 4, 1997. <http://margo.student.utwente.nl/simon/finished/thesis/thesis1/node6.html>

[4] Arkin, p 40.

[5] Arkin, p 52.

[6] James Messer, *Secrets of Network Cartography: A Comprehensive Guide to nmap*. NetworkUptime.com Publication, 11 Jan. 2006. <http://www.networkuptime.com/nmap/index.shtml>

[7] Arkin, p 60.

[8] Information Sciences Institute. *Transmission Control Protocol.* University of Southern California, Marina del Rey, California, September 1981, p 15-17.

[9] Information Sciences Institute, p 12.

[10] Information Sciences Institute, p 27-28.

[11] Information Sciences Institute, p 31.

[12] Information Sciences Institute, p 36.

[13] J. Postel. *RFC 768 User Datagram Protocol*. ISI, August 28, 1980. <http://www.networksorcery.com/enp/rfc/rfc768.txt>

[14] R. Braden. *RFC 1122 Requirements for Internet Hosts – Communication Layers*. Internet Engineering Task Force, October 1989. <http://www.networksorcery.com/enp/rfc/rfc1122.txt>

[15] *Internet Scanner User Guide Version 7.0, Service Pack 2*. Internet Security Systems, February 28, 2005, p 182. <http://documents.iss.net/literature/InternetScanner/IS_UG_7.0_SP2.pdf>

[16] *Nmap Reference Guide (Man Page)*. Dec. 13, 2005. INSECURE.ORG. January 11, 2006. <http://www.insecure.org/nmap/man/>

[17] Balaji Ganesan. *TCP/IP Stack Fingerprinting for Patch Detection in a Distributed Windows Environment.* Lane Department of Computer Science and Electrical Engineering, Morgantown, West Virginia, 2004.

[18] Robin Whaling. *OS Identification Methods and Countermeasures.* Foundstone Inc., August 2003. <http://www.foundstone.com/resources/whitepapers/wp_os_id_methods.pdf>

[19] Protocol Numbers. Internet Assigned Numbers Authority, September 2005. <http://www.iana.org/assignments/protocol-numbers >

[20] J. Treurniet. *An Overview of Passive Information Gathering Techniques for Network Security*, TM 2004-073. Defense R&D Canada – Ottawa. Ottawa, Ontario, May 2004. <http://www.ottawa.drdc-rddc.gc.ca/docs/e/TM2004-073.pdf>