Chapter 1: The World of Network Analysis

Wireshark Certified Network Analyst Exam Objectives covered:

- Defining Network Analysis
- Troubleshooting Tasks for the Network Analyst
- Security Tasks for the Network Analyst
- Optimization Tasks for the Network Analyst
- Application Analysis Tasks for the Network Analyst
- Understand Security Issues Related to Network Analysis
- Be Aware of Legal Issues of Listening to Network Traffic
- Overcome the “Needle in the Haystack Issue”
- Review a Checklist of Analysis Tasks

- Case Study: Pruning the “Puke”
- Case Study: The “Securely Invisible” Network
- Summary
- Practice What You’ve Learned
- Review Questions and Answers

Defining Network Analysis

Network analysis is the process of listening to and analyzing network traffic. Network analysis offers an insight into network communications to identify performance problems, locate security breaches, analyze application behavior, and perform capacity planning. Network analysis (aka "protocol analysis") is a process used by IT professionals who are responsible for network performance and security.

Whether you are completely new to network analysis or just returning after a hiatus of setting up servers, architecting your company's security plan, deploying Voice over IP, or jumping through hoops to get WLAN issues fixed… *Welcome and welcome back!*

Network analysis is not brain surgery. Anyone can analyze network communications. You do, however, need to acquire three basic skills to be a top notch network analyst who can spot the cause of performance problems, evidence of breached hosts, misbehaving applications or the impending overload of the network.

1. A solid understanding of TCP/IP communications
2. Comfort using Wireshark
3. Familiarity with packet structures and typical packet flows

Many of you have probably installed and configured TCP/IP networks—in fact, I imagine many of you have set up hundreds if not thousands of TCP/IP clients and servers. Excellent! You already understand TCP/IP addressing and realize the role that DNS and DHCP servers play on your network.

From a network analyst's perspective, you need to understand the purpose of those devices and protocols and how they interact. For example, how exactly does a DHCP server offer an IP address and configuration information to a DHCP client? What if there is a relay agent in use? What happens when the user's address lease time expires? How does the user learn the right IP address to use when the user wants to reach *www.wireshark.org*? What happens if the local name server does not have the answer? What happens if the local name server is down?

Seeing these processes in action at packet level is a fast way to learn the inner workings of your network. You build your baseline of understanding—the baseline is your foundational knowledge of how the processes are supposed to work.

Network analyzer tools are often referred to as "sniffers" and may be sold or distributed as a hardware-plus-software solution or as a software-only solution. Wireshark is distributed as an open source software-only solution, but there are add-on adapters that can enhance Wireshark's capabilities. The AirPcap adapter from CACE Technologies is an example of a hardware add-on. The AirPcap adapter is used on Windows hosts running Wireshark to listen in to wireless traffic in Monitor Mode.[5]

[5] Monitor Mode (also referred to as *rfmon* mode and wireless network analysis is covered in *Chapter 26: Introduction to 802.11 (WLAN) Analysis*.

Follow an Analysis Example

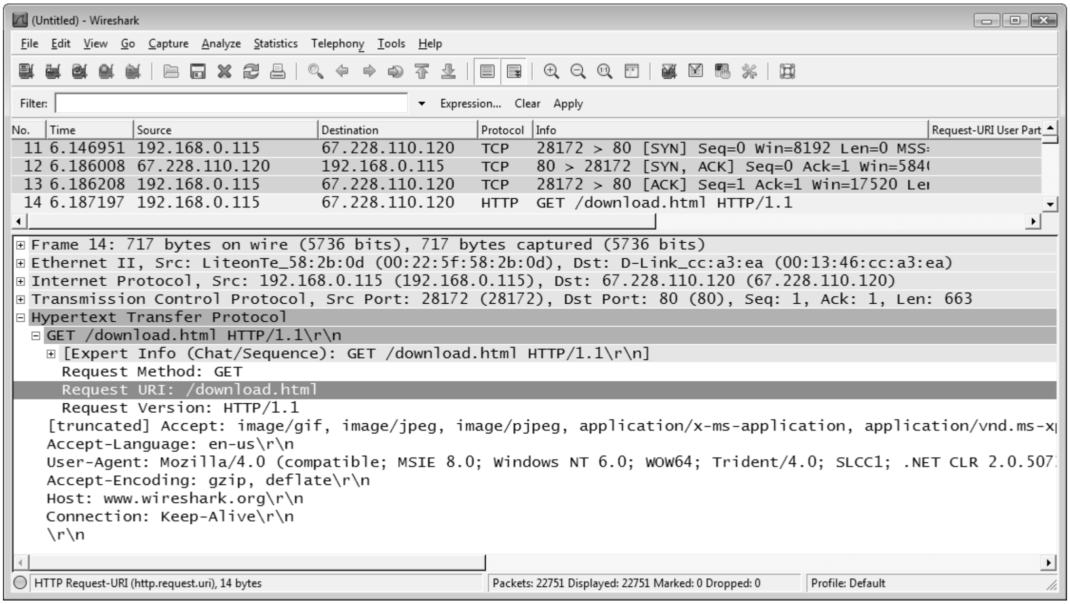
The typical network analysis session includes several tasks:

- Capture packets at the appropriate location
- Apply filters to focus on traffic of interest
- Review and identify anomalies in the traffic

For example, watch your own traffic as you browse to *www.wireshark.org/download.html* to grab the latest copy of Wireshark. This is what you might see in the traffic…

Your system requests the MAC (hardware) address of a local DNS server before asking for the IP address for *www.wireshark.org*. Hopefully, the DNS server responds with the information you need and then you're off!

Your client makes a TCP connection to *www.wireshark.org* and then sends an HTTP GET request asking for the download page as shown in Figure 1.


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply

```
No. Time     Source          Destination     Protocol Info                                                 Request-URI User Part
 11 6.146951 192.168.0.115   67.228.110.120  TCP      28172 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS
 12 6.186008 67.228.110.120  192.168.0.115   TCP      80 > 28172 [SYN, ACK] Seq=0 Ack=1 Win=584
 13 6.186208 192.168.0.115   67.228.110.120  TCP      28172 > 80 [ACK] Seq=1 Ack=1 Win=17520 Le
 14 6.187197 192.168.0.115   67.228.110.120  HTTP     GET /download.html HTTP/1.1
```

```
⊞ Frame 14: 717 bytes on wire (5736 bits), 717 bytes captured (5736 bits)
⊞ Ethernet II, Src: LiteonTe_58:2b:0d (00:22:5f:58:2b:0d), Dst: D-Link_cc:a3:ea (00:13:46:cc:a3:ea)
⊞ Internet Protocol, Src: 192.168.0.115 (192.168.0.115), Dst: 67.228.110.120 (67.228.110.120)
⊞ Transmission Control Protocol, Src Port: 28172 (28172), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 663
⊟ Hypertext Transfer Protocol
  ⊟ GET /download.html HTTP/1.1\r\n
    ⊞ [Expert Info (Chat/Sequence): GET /download.html HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /download.html
      Request Version: HTTP/1.1
    [truncated] Accept: image/gif, image/jpeg, image/pjpeg, application/x-ms-application, application/vnd.ms-x
    Accept-Language: en-us\r\n
    User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; WOW64; Trident/4.0; SLCC1; .NET CLR 2.0.507
    Accept-Encoding: gzip, deflate\r\n
    Host: www.wireshark.org\r\n
    Connection: Keep-Alive\r\n
    \r\n
```

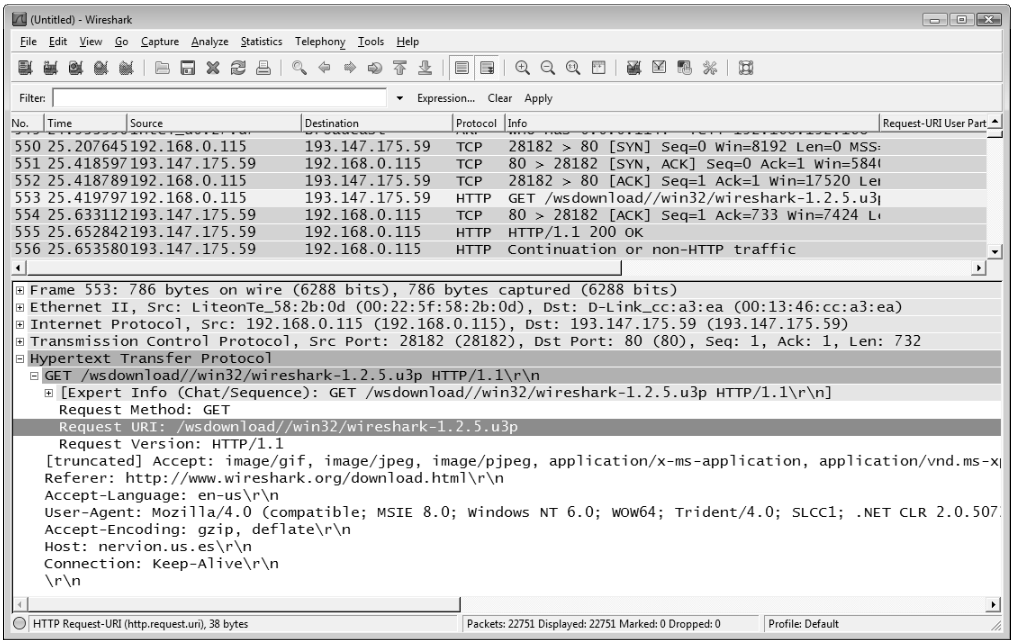
HTTP Request-URI (http.request.uri), 14 bytes | Packets: 22751 Displayed: 22751 Marked: 0 Dropped: 0 | Profile: Default

Figure 1. The client requests the Wireshark download.html page

If all goes well up to this point, you will see the HTTP server respond with an HTTP/1.1 200 OK response and then the page download begins. You will see various GET requests sent from your system—you are requesting the style sheets for the page and graphics to build the page.

When you select to download Wireshark, you see your system make a new TCP connection to another IP address and send a GET request for the Wireshark software as shown in Figure 2.

So far everything makes sense. You located the Wireshark site. You asked for the download.html page. Now you are downloading the file you want.



```
No. Time     Source          Destination     Protocol Info                                         Request-URI User Part
550 25.207645192.168.0.115   193.147.175.59  TCP      28182 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS:
551 25.418597193.147.175.59  192.168.0.115   TCP      80 > 28182 [SYN, ACK] Seq=0 Ack=1 Win=584
552 25.418789192.168.0.115   193.147.175.59  TCP      28182 > 80 [ACK] Seq=1 Ack=1 Win=17520 Le
553 25.419797192.168.0.115   193.147.175.59  HTTP     GET /wsdownload//win32/wireshark-1.2.5.u3
554 25.633112193.147.175.59  192.168.0.115   TCP      80 > 28182 [ACK] Seq=1 Ack=733 Win=7424 L
555 25.652842193.147.175.59  192.168.0.115   HTTP     HTTP/1.1 200 OK
556 25.653580193.147.175.59  192.168.0.115   HTTP     Continuation or non-HTTP traffic
```

```
⊞ Frame 553: 786 bytes on wire (6288 bits), 786 bytes captured (6288 bits)
⊞ Ethernet II, Src: LiteonTe_58:2b:0d (00:22:5f:58:2b:0d), Dst: D-Link_cc:a3:ea (00:13:46:cc:a3:ea)
⊞ Internet Protocol, Src: 192.168.0.115 (192.168.0.115), Dst: 193.147.175.59 (193.147.175.59)
⊞ Transmission Control Protocol, Src Port: 28182 (28182), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 732
⊟ Hypertext Transfer Protocol
  ⊟ GET /wsdownload//win32/wireshark-1.2.5.u3p HTTP/1.1\r\n
    ⊞ [Expert Info (Chat/Sequence): GET /wsdownload//win32/wireshark-1.2.5.u3p HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /wsdownload//win32/wireshark-1.2.5.u3p
      Request Version: HTTP/1.1
    [truncated] Accept: image/gif, image/jpeg, image/pjpeg, application/x-ms-application, application/vnd.ms-x
    Referer: http://www.wireshark.org/download.html\r\n
    Accept-Language: en-us\r\n
    User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; WOW64; Trident/4.0; SLCC1; .NET CLR 2.0.507
    Accept-Encoding: gzip, deflate\r\n
    Host: nervion.us.es\r\n
    Connection: Keep-Alive\r\n
    \r\n
```

```
○ HTTP Request-URI (http.request.uri), 38 bytes | Packets: 22751 Displayed: 22751 Marked: 0 Dropped: 0 | Profile: Default
```

Figure 2. You request the Wireshark executable

You can watch the process as the file is transferred to your local system. It all makes perfect sense. It is all quite logical.

Until it all goes “to hell in a hand basket” as my mother would say.

You sit patiently waiting for the download to finish—tapping your fingers ever so irritatingly on your desk. Your eyes wander… looking for some distraction that will make the time pass more quickly. Waiting… waiting… waiting… until finally you just can’t stand it anymore.

You type a new URL and decide to come back to the *www.wireshark.org* site later to get the latest copy of Wireshark. The other site loads quickly (oh… yeah… speed is good). You find another open source software package that is on your ‘must have’ list. You begin the download process and are filled with excitement at the thrill of taking charge and grabbing software at blazing speed (after all, your company did pay big money to upgrade that Internet connection)… until…

Your heart sinks…

This is taking waaaaay too long. At this rate you will miss lunch, dinner and potentially your summer vacation!

Maybe it's not *www.wireshark.org* that's having the problem. Maybe it's the Internet or (gasp!) your WAN link or (heaven's forbid!) your network or (shivers!) your DNS server or (unthinkable!) your desktop system.

Well? Which is it?

If you'd been running Wireshark in the background, you'd have known the answer long before I typed in that comment about your summer vacation. The packets never lie. They always point to where the problem is.

Network analysis adds an indispensible tool to the network—just as an x-ray is an indispensible tool to the hospital emergency room.[6]

Network analysis allows us the opportunity to look inside the network communication system. We can pull back the curtains and watch the packets travel back and forth. We can **SEE** the DNS query being sent out and catch the timely DNS response providing an answer. We can watch our local system send a TCP connection request packet to *www.wireshark.org*. We can measure how long it takes *www.wireshark.org* to answer and get a general feel for the round trip time to get to that site. We proudly beam as our system sends the HTTP GET request for the file—just as a good system should. We can gleefully…uh… what's that? The data transfer just stopped? How rude! Why did the transfer stop?

Well… you'll just have to look at the packets to know the answer. Then you can point the finger! In the world of finger pointing, ***it's only the network analyst's finger that counts***.

The scenario above is quite common. The page loads nicely—all those little pieces and parts zipping on down to your system, gently placed in your TCP receive buffer awaiting pick-up by the browser you used to download the file. Waiting… waiting… waiting… You are not the only one who is wondering what the *#$)(#! is taking so long. Your packets could wither and die in that buffer waiting to be picked up by the browser.

Wait… more data is coming in… and more… and… *SCREECH!* The TCP buffer is full and it kindly tells the *www.wireshark.org* server that it can't possibly handle another byte so please "shut up!" The file transfer stops. The buffer waits for the application to pick up data. The user waits to see the file download is complete. Everyone waits…

Where is that <insert "*bleep*" here> browser? Doesn't it have any sense of time? There's data down here to be delivered to the user. Someone's gonna get mad! Oh yeah—someone is mad.

In this case the problem is caused by a browser that is not picking up data out of the TCP receive buffer in a timely manner. The TCP receive buffer fills. The TCP stack at the client sends a TCP packet to the server to let it know that there is no more room to buffer data. The server stops sending the file until the client indicates there is more buffer space.

[6] Imagine if you took a bad fall ice skating (computer geeks should not ice skate—that's another story). You think you broke your arm. At the emergency room the doctors huddle around you perplexed. "It's probably just a sprain—a pain killer and no movement for a week and you'll be fine." chimes in one doctor. "No. I think it's broken—let's re-break it and set it" Eeek… this scenario gets even uglier when you consider appendicitis.

The problem is not at *www.wireshark.org*. The problem is not at the download server. The problem is not at the WLAN link. The problem is not the internal network. The problem IS at the client—specifically the browser. You know where to begin troubleshooting. In this book we cover several reasons why file transfers slow to a crawl.

Troubleshooting Tasks for the Network Analyst

Troubleshooting is the most common use of Wireshark and is performed to locate the source of unacceptable performance of the network, an application, a host or other element of network communications. Troubleshooting tasks that can be performed with Wireshark include, but are not limited to:

- Locate faulty network devices
- Identify device or software misconfigurations
- Measure high delays along a path
- Locate the point of packet loss
- Identify network errors and service refusals
- Graph queuing delays

Security Tasks for the Network Analyst

Security tasks can be both proactive and reactive and are performed to identify security scanning processes, holes or breaches on the network. Security tasks that can be performed with Wireshark include, but are not limited to:

- Perform intrusion detection
- Identify and define malicious traffic signatures
- Passively discover hosts, operating systems and services
- Log traffic for forensics examination
- Capture traffic as evidence
- Test firewall blocking
- Validate secure login and data traversal

Optimization Tasks for the Network Analyst

Optimization is the process of contrasting current performance with performance capabilities and making adjustments in an effort to reach optimal performance levels. Optimization tasks that can be performed with Wireshark include, but are not limited to:

- Analyzing current bandwidth usage
- Evaluating efficient use of packet sizes in data transfer applications
- Evaluating response times across a network

Application Analysis Tasks for the Network Analyst

Application analysis is the process of capturing and analyzing the traffic generated by a network application. Application analysis tasks that can be performed with Wireshark include, but are not limited to:

- Analyzing application bandwidth requirements
- Identifying application protocols and ports in use
- Validating secure application data traversal

Understand Security Issues Related to Network Analysis

Network analysis can be used to improve network performance and security—but it can also be used for malicious tasks. For example, an intruder who can access the network medium (wired or wireless) can listen in on traffic. Unencrypted communications (such as clear text user names and passwords) may be captured and thus enable a malicious user to compromise accounts. An intruder can also learn network configuration information by listening to the traffic—this information can then be used to exploit network vulnerabilities. Malicious programs may include network analysis capabilities to sniff the traffic.

Define Policies Regarding Network Analysis

Companies should define specific policies regarding the use of a network analyzer. Your company policies should state who can use a network analyzer on the network and how, when and where the network analyzer may be used. Ensure these policies are well known throughout the company.

If you are a consultant performing network analysis services for a customer, consider adding a “Network Analysis” clause to your non-disclosure agreement. Define network analysis tasks and be completely forthcoming about the types of traffic that network analyzers can capture and view.

Files Containing Network Traffic Should be Secured

Ensure you have a secure storage solution for the traffic that you capture because confidential information may exist in the traffic files (referred to as *trace files*).

Protect Your Network against Unwanted “Sniffers”

As you will learn in *Chapter 3: Capture Traffic*, switches make network analysis a bit more challenging. Those challenges can be overcome using taps or redirection methods. Switches are not security devices. Unused network ports and network ports in common areas (such as building lobbies) should be deactivated to discourage visitors from plugging in and listening to network traffic.

The best protection mechanism against network sniffing is to encrypt network traffic using a robust encryption method. Encryption solutions will not protect the general network traffic that is broadcast onto the network for device and/or service discovery however. For example, DHCP clients broadcast DHCP Discover and Request packets on the network. These packets contain information about the client (including the host name, requested IP address and other revealing information). These DHCP broadcasts will be forwarded out by all ports of a switch. A network analyzer connected to that switch is able to capture the traffic and learn information about the DHCP client.

Be Aware of Legal Issues of Listening to Network Traffic

We aren’t lawyers, so consult your legal counsel on this issue.

In general, Wireshark provides the ability to eavesdrop on network communications—have you heard the terms “wiretapping” or “electronic surveillance”? Unauthorized use of Wireshark may be illegal. Certain exceptions are in place to cover government use of wiretapping methods in advance of a crime being perpetrated.

In the U.S., Title I of the ECPA (Electronic Communications and Privacy Act), often referred to as the Wiretap Act, prohibits the intentional, actual or attempted interception, use, disclosure, or “procure[ment] [of] any other person to intercept or endeavor to intercept any wire, oral, or electronic communication.”

Title I offers exceptions for operators and service providers for uses “in the normal course of his employment while engaged in any activity which is a necessary incident to the rendition of his service” and for “persons authorized by law to intercept wire, oral, or electronic communications or to conduct electronic surveillance, as defined in section 101 of the Foreign Intelligence Surveillance Act (FISA) of 1978.” Cornell University Law School provides details of Title I at *www.law.cornell.edu/uscode/18/usc_sup_01_18_10_I_20_119.html*.

In the European Union, the Data Protection Directive, Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995, defines the protection of individuals with regard to the processing of personal data and on the free movement of such data requires Member States to ensure the rights and freedoms of natural persons with regard to the processing of personal data, and in particular their right to privacy, in order to ensure the free flow of personal data in the Community. For details on the EU Data Protection Directive, visit *ec.europa.eu/justice_home/fsj/privacy/*.

Avoid Prison

Company policies may also forbid unauthorized tapping into network communications. Disregard for these policies may result in disciplinary actions or termination.Tom Quilty, CEO of BD Consulting and Investigations (www.bdcon.net), offered this note:

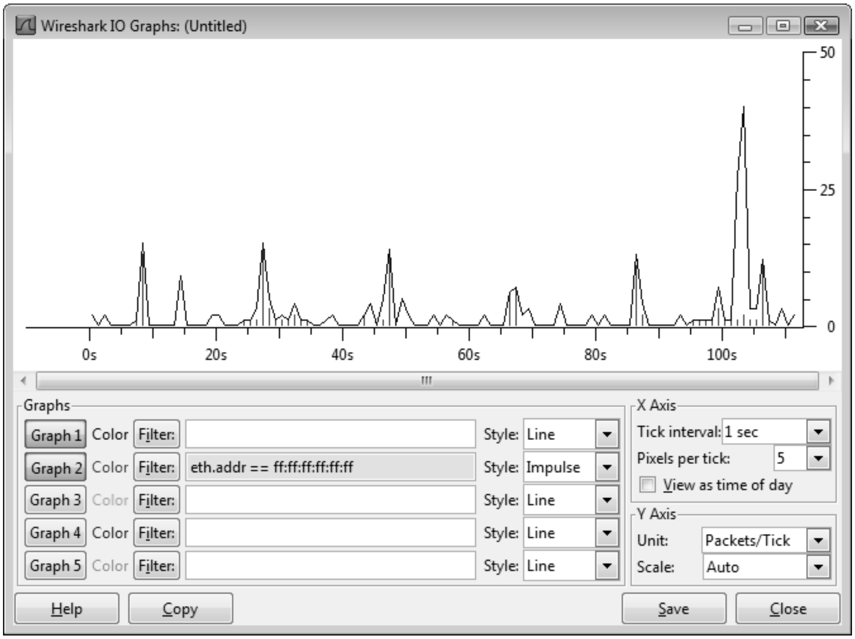
“If they are capturing traffic with Personally Identifiable Information (PII), HIPAA (health records), or other protected information, the trace files should not leave the facility. If lost, it may require that the client report a data breach, which could be very costly for the person capturing the traffic. They should also ensure that they have an appropriate General Liability and Errors & Omissions rider. I would recommend that they understand what information is going across the wire (or air) and review the client’s Data Breach Policies and Response Plan (assuming they have one—most don’t). They may also have to testify about how they protected any information captured (hopefully, they have developed procedures for this before this comes up).”

Many countries have similar laws in place regarding protection of information—make sure you understand your local laws and look into professional insurance… just in case.

Overcome the "Needle in the Haystack Issue"

Many times new analysts capture thousands (or millions) of packets and are faced with the "needle in a haystack issue"—the feeling that they are drowning in packets. Several non-pharmaceutical analysis procedures can be used to avoid or deal with this situation:

- Place the analyzer appropriately (covered in *Chapter 3: Capture Traffic*)
- Apply capture filters to reduce the number of packets captured (covered in *Chapter 4: Create and Apply Capture Filters*)
- Apply display filters to focus on specific conversations, connections, protocols or applications (covered in *Chapter 9: Create and Apply Display Filters*)
- Colorize the conversations in more complex multi-connection communications (covered in *Chapter 6: Colorize Traffic*)
- Reassemble streams for a clear view of data exchanged (covered in *Chapter 10: Follow Streams and Reassemble Data*)
- Save subsets of the captured traffic into separate files (covered in *Chapter 12: Save, Export and Print Packets*)
- Build graphs depicting overall traffic patterns or apply filters to graphs to focus on particular traffic types as shown in Figure 3 (covered in *Chapter 8: Interpret Basic Trace File Statistics* and *Chapter 21: Graph IO Rates and TCP Trends*).



Graph 1 Filter: Style: Line
Graph 2 Filter: eth.addr == ff:ff:ff:ff:ff:ff Style: Impulse
Graph 3 Filter: Style: Line
Graph 4 Filter: Style: Line
Graph 5 Filter: Style: Line

X Axis
Tick interval: 1 sec
Pixels per tick: 5
□ View as time of day

Y Axis
Unit: Packets/Tick
Scale: Auto

Figure 3. Use filters in graphs to identify traffic patterns

Throughout this book we will show and work with trace files obtained and manipulated using these techniques.

Review a Checklist of Analysis Tasks

Analysis tasks can be considered preventive or reactive. Preventive methods include baselining network communications to learn the current status of the network and application performance. Preventive analysis can also be used to spot network problems before they are felt by the network users. For example, identifying the cause of packet loss before it becomes excessive and affects network communications helps avoid problems before they are even noticed.

Reactive analysis techniques are employed after a complaint about network performance has been reported or when network issues are suspected. Sadly, reactive analysis is more common.

The following lists some of the analysis tasks that can be performed using Wireshark:

- Find the top talkers on the network
- Identify the protocols and applications in use
- Determine the average packets per second rate and bytes per second rate of an application or all network traffic on a link
- List all hosts communicating
- Learn the packet lengths used by a data transfer application
- Recognize the most common connection problems
- Spot delays between client requests due to slow processing
- Locate misconfigured hosts
- Detect network or host congestion that is slowing down file transfers
- Identify asynchronous traffic prioritization
- Graph HTTP flows to examine website referrals rates
- Identify unusual scanning traffic on the network
- Quickly identify HTTP error responses indicating client and server problems
- Quickly identify VoIP error responses indicating client, server or global errors
- Build graphs to compare traffic behavior
- Graph application throughput and compare to overall link traffic seen
- Identify applications that do not encrypt traffic
- Play back VoIP conversations to hear the effects of various network problems on network traffic
- Perform passive operating system and application use detection
- Spot unusual protocols and unrecognized port number usage on the network
- Examine the start up process of hosts and applications on the network
- Identify average and unacceptable service response times (SRT)
- Graph intervals of periodic packet generation applications or protocols

Networks vary greatly in the traffic seen. The number and type of network analysis tasks you can perform depends on your network traffic characteristics.

Understand Network Traffic Flows

Let's start at the packet level by following a packet as it makes its way from one host to another. We'll start by looking at where we can capture the traffic (more in-depth information on capturing can be found in *Chapter 3: Capture Traffic*). We will examine how a packet is encapsulated, then stripped nearly naked by some high-priced router only to be re-encapsulated and sent on its way again just before hypothermia sets in. Let's chat about packets whizzing past switches so quickly there really isn't even time for a proper introduction. Then we peak at the effect that Quality of Service (QoS) has on our traffic and where devices and technology puff up their chests, whip out their badges and throw up roadblocks that make us fear for our little packet lives.

Switching Overview

Switches are considered Layer 2 devices—a reference to Layer 2 of the Open Systems Interconnection (OSI) model—the data link layer which includes the Media Access Control (MAC) portion of the packet, such as the Ethernet header.

Switches forward packets based on the destination MAC address (aka the destination hardware address) contained in the MAC header. As shown in Figure 4, switches do not change the MAC or IP addresses in packets.[7]

When a packet arrives at a switch, the switch checks the packet to ensure it has the correct checksum. If the packet's checksum is incorrect, the packet is considered "bad" and the packet is discarded. Switches should maintain error counters to indicate how many packets they have discarded because of bad checksums.

If the checksum is good, the switch examines the destination MAC address of the packet and consults its MAC address table to determine if it knows which switch port leads to the host using that MAC address. If the switch does not have the target MAC address in its tables, it will forward the packet out all ports in hopes of discovering the target when it answers.

If the switch does have the target MAC address in its tables it forwards the packet out the appropriate port. Broadcasts are forwarded out all ports on a switch. Unless configured otherwise, multicasts are also forwarded out all ports on a switch.

To learn about the challenges of and solutions for capturing traffic on a switched network, refer to *Capture Traffic on Switched Networks* on page 82.

[7] In Figure 4 we use a symbolic letter to represent the MAC addresses of the client and server.

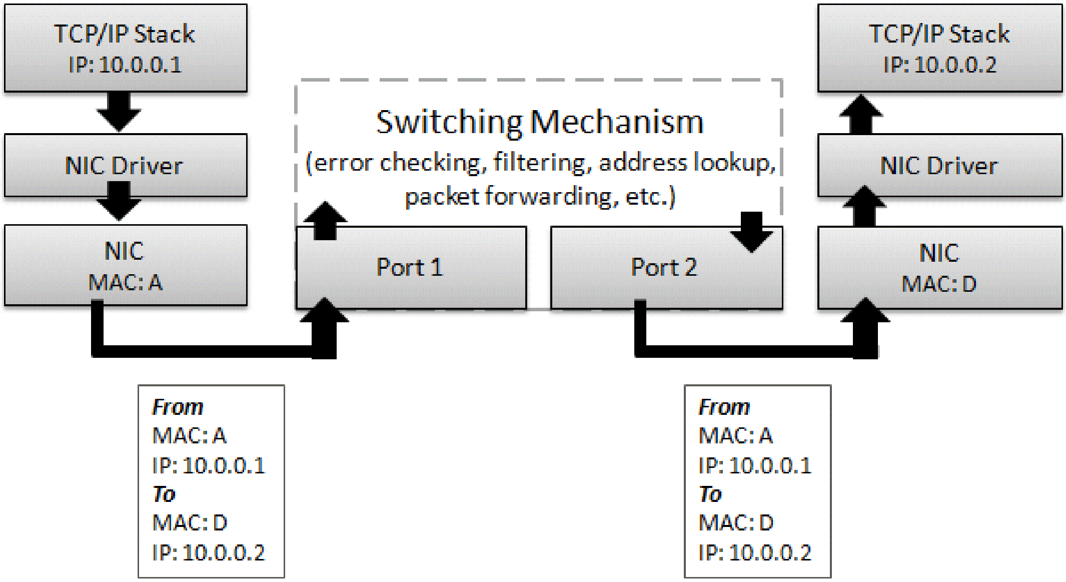


Figure 4. Switches do not alter the MAC or IP address in a packet

Routing Overview

Routers forward packets based on the destination IP address in the IP header. When a packet is sent to the MAC address of the router, that router examines the checksum to ensure the packet is valid. If the checksum is invalid, the packet is dropped. If the checksum is valid, the router strips off the MAC header (such as the Ethernet header) and examines the IP header to identify the “age” (in Time to Live) and destination of the packet. If the packet is too “old” (Time to Live value of 1), the router discards the packet.

The router consults its routing tables to determine if the destination IP network is known. If the router is directly connected to the target network, it can send the packet on to the target. The router decrements the IP header Time to Live value and then creates and applies a new MAC header on the packet before forwarding it, as shown in Figure 5.

If the target is not on a locally connected network, the router forwards the packet to the next-hop router that it learned about when consulting its routing tables.

Routers may contain rules that block or permit packets based on the addressing information. Many routers provide firewall capabilities and can block/permit traffic based on other characteristics.

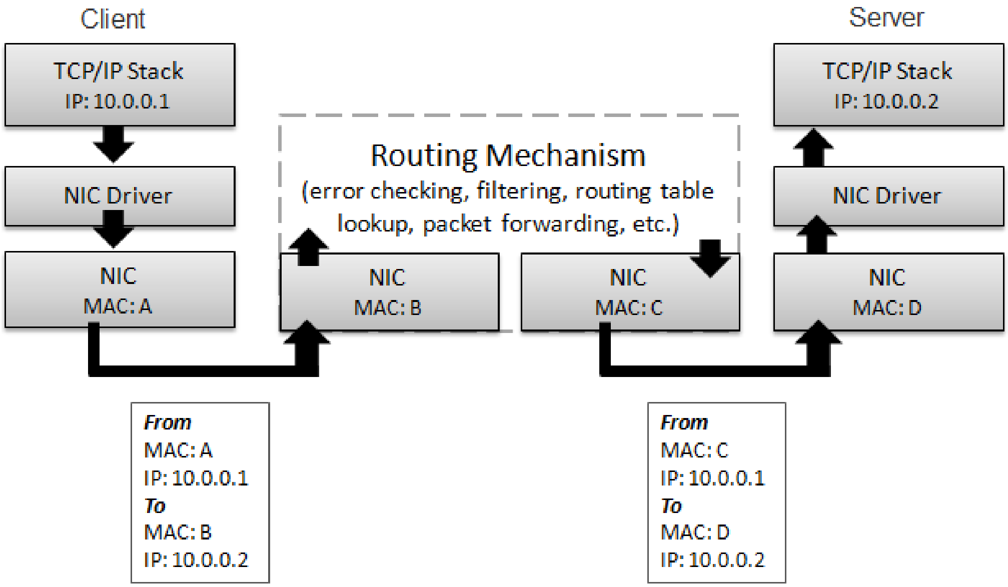


Figure 5. Routers change the destination MAC address to the target (if the target is local) or next router (if the target is remote)

Proxy, Firewall and NAT/PAT Overview

Firewalls are created to examine the traffic and allow/disallow communications based on a set of rules. For example, you may want to block all TCP connection attempts from hosts outside the firewall that are destined to port 21 on internal servers.

Basic firewalls operate at Layer 3 of the OSI model—the network layer. In this capacity, the firewall acts like a router when handling network traffic. The firewall will forward traffic that is not blocked by the firewall rules. The firewall prepends a new MAC header on the packet before forwarding it. Additional packet alteration will take place if the firewall supports added features, such as Network Address Translation (NAT) or proxy capabilities.

NAT systems alter the IP addresses in the packet as shown in Figure 6. This is often used to hide the client's private IP address. A basic NAT system simply alters the source and destination IP address of the packet and tracks the connection relationships in a table to forward traffic properly when a reply is received. Port Address Translation (PAT) systems also alter the port information and use this as a method for demultiplexing multiple internal connections when using a single outbound address. The IP addresses you see on one side of a NAT/PAT device will not match the IP addresses you see on the other side of the NAT/PAT device. To correlate the communications on both sides of a NAT device, you will need to look past the IP header to identify matching packets.

Proxy servers also affect traffic. Unlike the communications seen when you use a standard firewall, the client connects to the proxy server and the proxy server makes a separate connection to the target. There are two totally separate connections to examine when troubleshooting these communications.

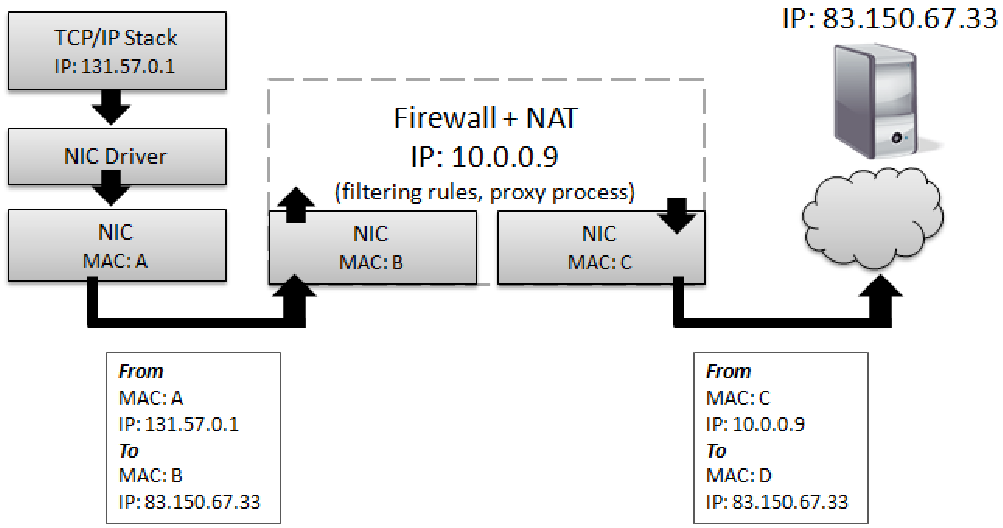


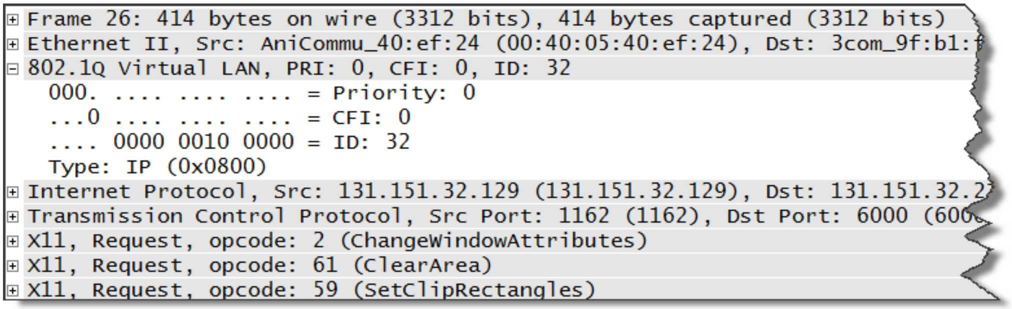
Figure 6. The firewall uses NAT to hide the true source IP address

Other Technologies that Affect Packets

There are numerous other technologies that affect network traffic patterns and packet contents.

Virtual LAN (VLAN) tagging (defined as 802.1Q) adds an identification (tag) to the packets. This tag is used to create virtual networks in a switched environment. Figure 7 shows a VLAN tag in an Ethernet frame. In this case, the sender belongs to VLAN 32.

Multiprotocol Label Switching (MPLS) is a method of creating virtual links between remote hosts. MPLS packets are prefaced with a special header by MPLS edge devices. For example, a packet sent from a client reaches an MPLS router where the MPLS label is placed on the packet. The packet is now forwarded based on the MPLS label, not routing table lookups. The MPLS label is stripped off when the packet exits the MPLS network.



```
⊞ Frame 26: 414 bytes on wire (3312 bits), 414 bytes captured (3312 bits)
⊞ Ethernet II, Src: AniCommu_40:ef:24 (00:40:05:40:ef:24), Dst: 3com_9f:b1:
⊟ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 32
    000. .... .... .... = Priority: 0
    ...0 .... .... .... = CFI: 0
    .... 0000 0010 0000 = ID: 32
    Type: IP (0x0800)
⊞ Internet Protocol, Src: 131.151.32.129 (131.151.32.129), Dst: 131.151.32.2
⊞ Transmission Control Protocol, Src Port: 1162 (1162), Dst Port: 6000 (600
⊞ X11, Request, opcode: 2 (ChangeWindowAttributes)
⊞ X11, Request, opcode: 61 (ClearArea)
⊞ X11, Request, opcode: 59 (SetClipRectangles)
```

Figure 7. VLAN tags separate virtual networks using the ID field

Warnings About "Smarter" Infrastructure Devices

You paid a bunch of money for those brilliant infrastructure devices and you didn't expect them to be the cause of your network problems, did you? Numerous "security devices" do more than route packets based on simple rules—they get in there and mess up the packets. For example, Cisco's Adaptive Security Appliance (ASA) performs "TCP normalization." Billed as stateful firewalls and VPN concentrators, these lovely boxes had a little problem that caused them to strip off some TCP functionality during the connection process. In essence, an ASA device forced TCP hosts on both sides of it to go back to pre-2006 capabilities.

Wide Area Network (WAN) optimization techniques can also alter the packet and data stream process by compressing traffic, offering locally-cached versions of data, optimizing TCP or prioritizing traffic based on defined characteristics (traffic "shaping").

The best way to know how these technologies affect your traffic is to capture the packets before and after they pass through a traffic-altering device.

Launch an Analysis Session

You can start capturing and analyzing traffic right now. Follow these steps to get a feel for analyzing traffic on a wired network first.

Step 1: Get Wireshark installed (refer to the System Requirement information at *www.wireshark.org/docs/wsug_html_chunked/ChIntroPlatforms.html*). Visit *www.wireshark.org/docs/wsug_html_chunked/ChapterBuildInstall.html* for details on installing Wireshark on numerous platforms.

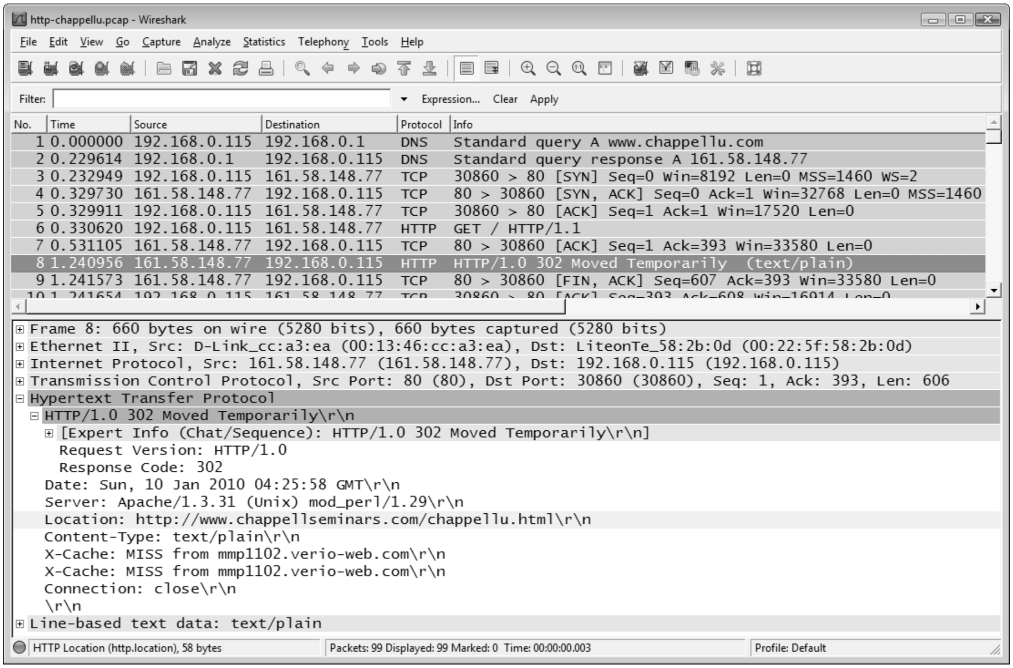
Step 2: Launch Wireshark and click on your wired network adapter listed in the Interface List on the Start Page. If your adapter is not listed, you cannot capture traffic. Visit *wiki.wireshark.org/CaptureSetup/NetworkInterfaces* for assistance. If your adapter was listed, Wireshark should be capturing traffic now.

Step 3: If you have browsed to *www.chappellU.com* recently, clear your browser cache before this step. Refer to your browser Help for details on how to clear your browser cache. In addition, consider clearing your DNS cache.[8] While Wireshark is capturing traffic, launch your browser and visit *www.chappellU.com*.

Step 4: Select **Capture | Stop** or click the **Stop Capture** button.

[8] To clear your DNS cache on a Windows host, go to a command prompt and type `ipconfig /flushdns`. On a Linux host, restart the `nscd` (name service cache) daemon. For MAC OS X 10.5.x or 10.6.x, type `dscacheutil -flushcache` at the terminal prompt.

Step 5: Look through the captured traffic. You should see a DNS query (unless you did not clear your DNS cache in Step 3). After you make a connection to *www.chappellU.com*, your browser sent a GET request to the server and should have received an HTTP/1.0 302 Moved Temporarily response as shown in Figure 8. You are redirected to *www.chappellseminars.com/chappellu.html*.


http-chappellu.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply

```
No. Time      Source         Destination    Protocol Info
  1 0.000000  192.168.0.115  192.168.0.1    DNS      Standard query A www.chappellu.com
  2 0.229614  192.168.0.1    192.168.0.115  DNS      Standard query response A 161.58.148.77
  3 0.232949  192.168.0.115  161.58.148.77  TCP      30860 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
  4 0.329730  161.58.148.77  192.168.0.115  TCP      80 > 30860 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
  5 0.329911  192.168.0.115  161.58.148.77  TCP      30860 > 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
  6 0.330620  192.168.0.115  161.58.148.77  HTTP     GET / HTTP/1.1
  7 0.531105  161.58.148.77  192.168.0.115  TCP      80 > 30860 [ACK] Seq=1 Ack=393 Win=33580 Len=0
  8 1.240956  161.58.148.77  192.168.0.115  HTTP     HTTP/1.0 302 Moved Temporarily  (text/plain)
  9 1.241573  161.58.148.77  192.168.0.115  TCP      80 > 30860 [FIN, ACK] Seq=607 Ack=393 Win=33580 Len=0
 10 1.241654  192.168.0.115  161.58.148.77  TCP      30860 > 80 [ACK] Seq=393 Ack=608 Win=16914 Len=0
```

```
⊞ Frame 8: 660 bytes on wire (5280 bits), 660 bytes captured (5280 bits)
⊞ Ethernet II, Src: D-Link_cc:a3:ea (00:13:46:cc:a3:ea), Dst: LiteonTe_58:2b:0d (00:22:5f:58:2b:0d)
⊞ Internet Protocol, Src: 161.58.148.77 (161.58.148.77), Dst: 192.168.0.115 (192.168.0.115)
⊞ Transmission Control Protocol, Src Port: 80 (80), Dst Port: 30860 (30860), Seq: 1, Ack: 393, Len: 606
⊟ Hypertext Transfer Protocol
  ⊟ HTTP/1.0 302 Moved Temporarily\r\n
    ⊞ [Expert Info (Chat/Sequence): HTTP/1.0 302 Moved Temporarily\r\n]
      Request Version: HTTP/1.0
      Response Code: 302
    Date: Sun, 10 Jan 2010 04:25:58 GMT\r\n
    Server: Apache/1.3.31 (Unix) mod_perl/1.29\r\n
    Location: http://www.chappellseminars.com/chappellu.html\r\n
    Content-Type: text/plain\r\n
    X-Cache: MISS from mmp1102.verio-web.com\r\n
    X-Cache: MISS from mmp1102.verio-web.com\r\n
    Connection: close\r\n
    \r\n
⊞ Line-based text data: text/plain
```

```
HTTP Location (http.location), 58 bytes    Packets: 99 Displayed: 99 Marked: 0  Time: 00:00:00.003    Profile: Default
```

Figure 8. The HTTP server indicates that a page has been moved

You may see traffic from other processes in the trace file. For example, if your browser performs a website blacklist check to identify known malicious sites, you will see this traffic preceding connection to *www.chappellU.com*. Display filters can be used to remove unrelated traffic from view so you can focus better on the traffic of interest. For more information on display filtering, refer to *Chapter 9: Create and Apply Display Filters*.

Step 6: Select **File | Save** and create a *mytraces* directory. Save your file using the name *chappellu.pcap* (Wireshark automatically appends the .pcap extension if you forget to include it).

You did it! Well done. You are well on your way to learning network analysis—one of the most valuable and fundamental skills of network management and security.

Case Study: Pruning the "Puke"

Submitted by: ***Mitch Dickey***
Frederick County Public Schools, VA

Our school district is comprised of 24 buildings and roughly 50 VLANS. Generally speaking, each campus has one VLAN for data and one for voice. For the most part each campus is its own VLAN, with some smaller sites sharing a single VLAN. We operate in a NetWare environment with at least one NetWare server at each campus. Each campus links back to an aggregate location before it is sent on to the router; what some like to call "Router on a Stick."

I use Wireshark on a regular basis to monitor traffic patterns and remove unnecessary traffic from the VLANs that I manage. Two types of traffic that I have eliminated are NetBIOS and SMB. Since we are in a NetWare environment we use NDPS for our printing services and do not require Windows File and Printer Sharing. Because of this I turn off NetBIOS and SMB on the machines that I manage. I recently sampled four other VLANs (out of my control) by taking a five minute PCAP. After the capture, I sifted through the traffic using filters to determine what percentage of traffic was NetBIOS and SMB. Although the results are lower than what I expected, trimming what I did find could be beneficial to switch/router processing, and most of all security.

- A capture containing 50,898 packets returned a combined total of 1,321 packets or 2.5% NetBIOS/SMB traffic.
- A capture containing 175,824 packets returned a combined total of 16,480 packets or 9% NetBIOS/SMB traffic.
- A capture containing 295,911 packets returned a combined total of 14,102 packets or 5% NetBIOS/SMB traffic.
- A capture containing 115,814 packets returned a combined total of 333 packets or less than 1% NetBIOS/SMB traffic.

I have used Wireshark to track down and remove other unnecessary protocols like SNMP and SSDP as well. We only use SNMP on Cisco equipment so eliminating it from network printers has cleaned up the network.

Case Study:
The “Securely Invisible” Network

One customer’s network consisted of 22 buildings in a campus-style setting. Management complained that the network is slow at times and had asked a consultant to come onsite to determine the cause of poor network performance.

Upon arrival, I was asked to sign a legal document stating that I would not listen to the network traffic to isolate the problem (you, as I do, must question why they called me).

The management at this company was concerned that confidential data may traverse their network in an unencrypted form.

The management was ignoring the fact that there are many ways for someone to tap into their network. If their data is visible to a network analyst, it would be best to verify that and fix the problem, not just assume that no one is listening.

It took several meetings with various individuals to convince management that they were a bit “off” on their thinking.

Once I began listening to their network traffic it became evident that they had good reason to be concerned. Their Lotus Notes implementation was misconfigured—all emails traveled through the network in clear text.

Over the next few hours of listening to the network traffic we found several applications that sent sensitive data across the network. By the time I left they had a list of security enhancements to implement on the network.

Summary

Network analysis offers an insight into network communications. When performance problems plague the network, guesswork can often be time-consuming and lead to inaccurate conclusions costing you and your company time and money. A full understanding of the network traffic flows is necessary to (a) place the analyzer properly on the network and (b) identify possible causes of network problems.

At this point it is recommended that you follow the procedures listed in *Launch an Analysis Session* on page 15 and review the section entitled *Follow an Analysis Example* on page 3.

Practice What You've Learned

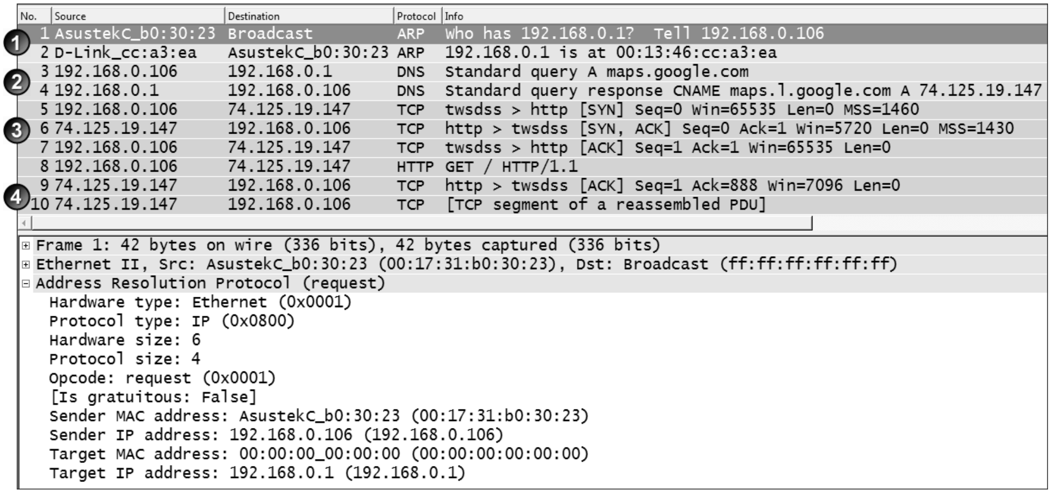
Download the trace files available in the Download section of the book website, *www.wiresharkbook.com*. There are many trace files and other book supplement files available on the book website. Consider copying them all to your drive now.

In Wireshark, open *gen-googlemaps.pcap*. This trace file contains the traffic from a web browsing session to *maps.google.com*.

Our client is 192.168.0.106. Our default gateway, 192.168.0.1, offers DNS services as well.

Answer the following questions about this trace file.

- What is the hardware address of the client that is browsing to *maps.google.com*?
- What is the IP address of the DNS server (which is also the router)?
- What is the hardware address of the DNS server/router?
- What IP addresses are associated with *maps.google.com*?



```
No. Source            Destination       Protocol Info
 1 AsustekC_b0:30:23 Broadcast         ARP   Who has 192.168.0.1?  Tell 192.168.0.106
 2 D-Link_cc:a3:ea   AsustekC_b0:30:23 ARP   192.168.0.1 is at 00:13:46:cc:a3:ea
 3 192.168.0.106     192.168.0.1       DNS   Standard query A maps.google.com
 4 192.168.0.1       192.168.0.106     DNS   Standard query response CNAME maps.l.google.com A 74.125.19.147
 5 192.168.0.106     74.125.19.147     TCP   twsdss > http [SYN] Seq=0 Win=65535 Len=0 MSS=1460
 6 74.125.19.147     192.168.0.106     TCP   http > twsdss [SYN, ACK] Seq=0 Ack=1 Win=5720 Len=0 MSS=1430
 7 192.168.0.106     74.125.19.147     TCP   twsdss > http [ACK] Seq=1 Ack=1 Win=65535 Len=0
 8 192.168.0.106     74.125.19.147     HTTP  GET / HTTP/1.1
 9 74.125.19.147     192.168.0.106     TCP   http > twsdss [ACK] Seq=1 Ack=888 Win=7096 Len=0
10 74.125.19.147     192.168.0.106     TCP   [TCP segment of a reassembled PDU]

⊞ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
⊞ Ethernet II, Src: AsustekC_b0:30:23 (00:17:31:b0:30:23), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
⊟ Address Resolution Protocol (request)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (0x0001)
    [Is gratuitous: False]
    Sender MAC address: AsustekC_b0:30:23 (00:17:31:b0:30:23)
    Sender IP address: 192.168.0.106 (192.168.0.106)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.0.1 (192.168.0.1)
```

❶ The first two packets—ARP packets—obtain the hardware address of the DNS server. What can we learn just from these two packets? Well—the client is 192.168.0.106. The DNS server is at 192.168.0.1. The hardware addresses of the client and the DNS server are listed in the Source and Destination columns (the first three bytes of the hardware address—the OUI value—and "broadcast" has been resolved to a more readable format by Wireshark). The hardware address of the client is listed as AsustekC_b0:30:23 in the Packet Info pane and 00:17:31:b0:30:23 in the Ethernet II summary line and inside the ARP packet.

❷ Packets 3 and 4 are the DNS query/response packets. The client is trying to get the IP address of *maps.google.com*. The DNS query packet is addressed to the hardware address and IP address of the DNS server (this DNS server is local to the client). The DNS server provides 7 IP addresses and indicates that *maps.google.com*'s real name (CNAME) is *maps.l.google.com*. The first address provided is 74.125.19.147.

❸ The client makes a TCP connection to *maps.google.com* in packets 5, 6 and 7. Now the client sends the packet to the hardware address of the router (which is also the DNS server) and the IP address of *maps.google.com* (*maps.l.google.com*).

❹ In packet 8 the client asks for the main page (GET / HTTP/1.1). In packet 9, the server acknowledges receipt of that request. In packet 10 the server begins sending the main page to the client.

The following table lists the trace file you worked with and a couple other trace files at *www.wiresharkbook.com* that you might want to review.

gen-googlemaps.pcap	This trace file depicts a simple web browsing session to *www.google.com*. The client performed an ARP query to get the hardware address of the DNS server and then sent a query to that DNS server to resolve the IP address for *www.google.com*. After receiving a successful response, the client makes a TCP connection to the server on port 80 and requests to GET the main page. The page is downloaded successfully.
telnet.pcap	Someone makes a telnet connection to a Cisco router to run the `show version` command which is echoed back, as is the `exit` command. The password, however, is not echoed back. Follow the DO, DON'T, WILL and WON'T command as the client and server negotiate the connection behavior.
icmp-ping-basic.pcap	This trace shows a basic ICMP ping test preceded by the DNS query/response to obtain the IP address of *www.chappellU.com*.

Review Questions

Q1.1 **What is the purpose of network analysis?**

Q1.2 **Name at least three troubleshooting tasks that can be performed using network analysis.**

1.

2.

3.

Q1.3 **Why is network analysis considered a security risk by some companies?**

Answers to Review Questions

Q1.1 **What is the purpose of network analysis?**

A1.1 Network analysis offers an insight into network communications to identify performance problems, locate security breaches, analyze application behavior, and perform capacity planning.

Q1.2 **Name at least three troubleshooting tasks that can be performed using network analysis.**

A1.2 **1.** Locate faulty network devices

2. Measure high delays along a path

3. Locate the point of packet loss

Q1.3 **Why is network analysis considered a security risk by some companies?**

A1.3 Some companies consider network analysis to be a security risk because it involves tapping into network traffic and eavesdropping on communications. These companies fear that unencrypted information (data, email, etc.) may be seen by the network analyst. In reality, however, the network analyst can identify unsecure network communications to prevent unauthorized eavesdroppers from gaining insight into confidential communications.