James Madison University

CS633 Project 4

Author:
Josh Feehs

April 17, 2014

1 Executive Summary

The task I was given was to analyze an approximately two-hour-long packet capture from acme's network and determine if an attack had occured, and what happened. Using a few different forensics tools, I was able to find that there was indeed an attack on the network that resulted in a malicious user gaining remote access to the company's web server and reading the secret.txt file that was stored on that server. I found that a machine that does not fit the normal workstation profile logged into the network for a limited amount of time and performed almost solely reconnaissance and a subsequent attack on the network. The technical details about how I set-up my workstation for analysis and the details of the attack are in the rest of the report.

It is important to note that this attack could have easily been prevented had more attention/importance been given to physical and computer security in the company. The attack that compromised the web server has been well-known for at least six to seven years before this attack occurred on the acme network. I have made a series of recommendations as to how this attack could have been prevented and how to prevent future, similar (or more dangerous) attacks. These recommendations are located in the Conclusions section at the end of my report.

2 Set-up for Analysis

The first action I had to take was to use sftp and copy the pcap file from the JMU server. I did this, copied the file over to my virtual machines, and verified the SHA256 checksum.

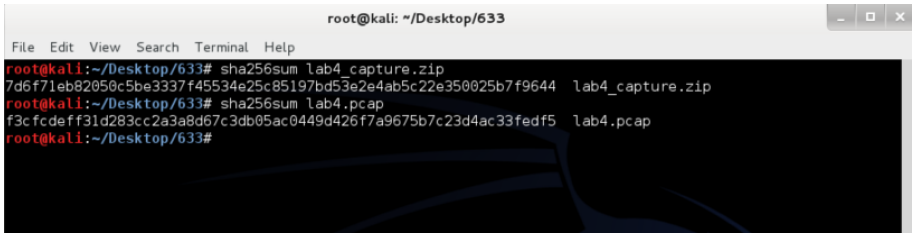


Figure 1: The verified SHA-256 checksum

In performing my analysis, I used two main tools. The main tool that I used was Wireshark. I used the default version of Wireshark that comes pre-installed in Kali linux. This was the main tool for my investigation. A second tool that I used is called Network Miner. This is a Windows tool that can perform static analysis on a pcap file and give a lot of useful output. I downloaded this tool from http://www.netresec.com/?page=NetworkMiner . Network Miner was especially useful to me in identifying the network topology and providing basic information about each of the endpoints and other machines on the network.

I used the conclusions that it made about each machine as a starting point in trying to make my own conclusions.

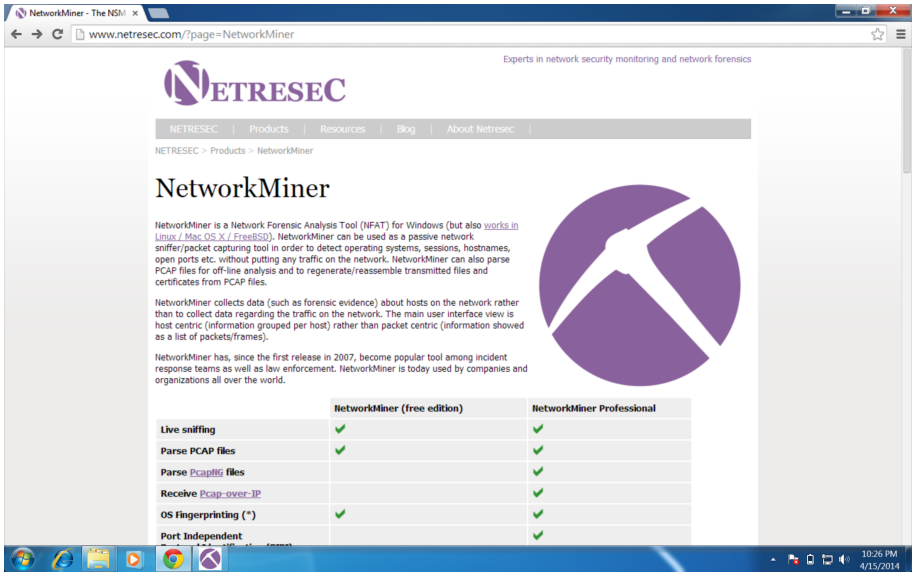


Figure 2: The web page where I downloaded network miner

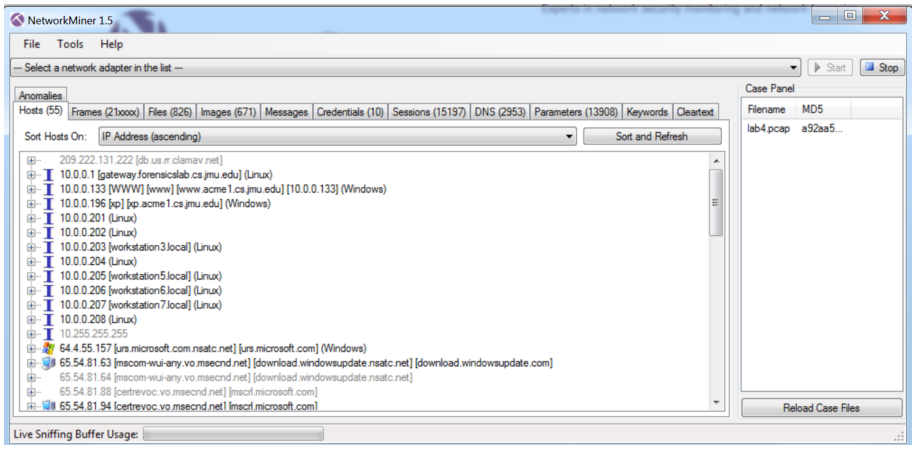


Figure 3: The main screen of network miner- this shows it identifying hosts

3 Network Topology

The following section is focused on describing the computers that were on the system that had network trafic captured. At the end, I discuss a few of the

remote hosts that appear in the traffic capture. The goal of this section is to accurately describe the type of machines that each machine is, including operating system and other general details. This section will not describe any of the machines as the victim or attacking machine.

3.1 10.0.0.1

Network Miner reports that the name of this machine is "gateway.forensicslab.cs.jmu.edu." Obviously, that suggests that this is the gateway machine for the network, which would fit with the general trend that the gateway has an ip adress of the subnet ending in .1. Digging deeper into traffic involving the machine, I found that 10.0.0.1 is the primary machine for receiving DNS and DHCP requests, and it answers those requests. Below is a screenshot showing some of those requests.

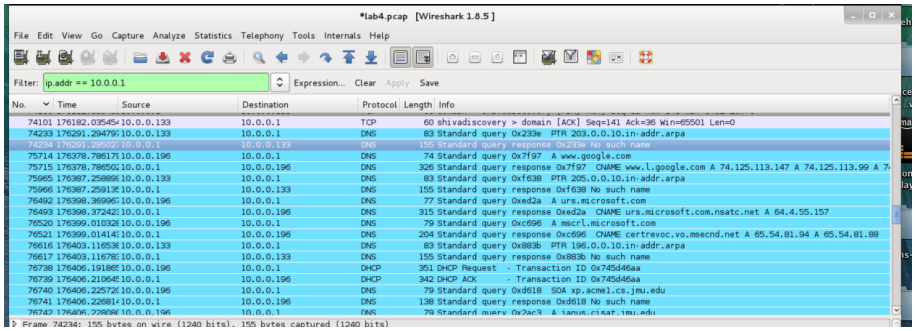


Figure 4: DNS and DHCP transactions involving 10.0.0.1

3.2 10.0.0.133

Using Network Miner, I discovered that the name of this machine is "WWW." Using this as a basis, I then opened the recovered index.html (recovered first using Network Miner, then again exporting it from wireshark), and it showed what is seen below:

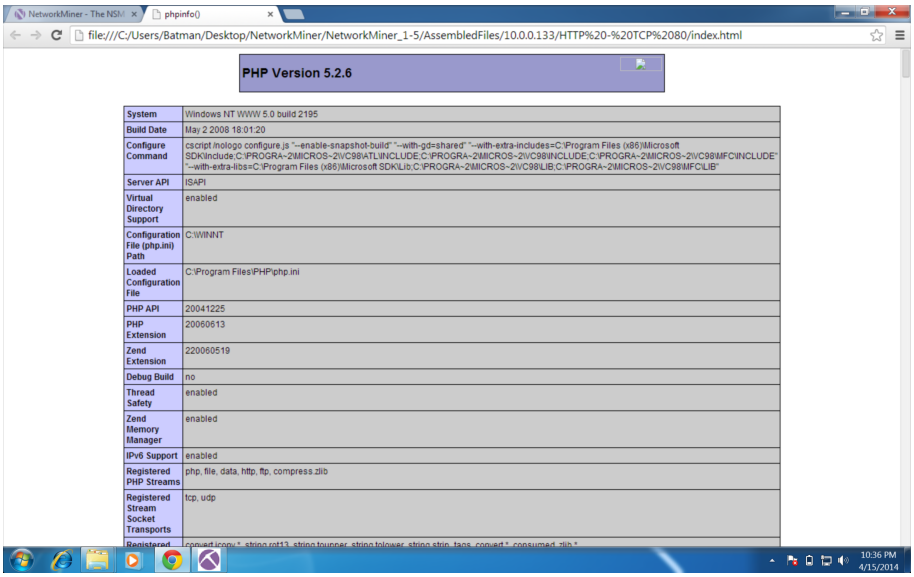


Figure 5: Index.html from 10.0.0.133

After doing a bit more research, I found that Windows NT 5.0 is another name for Windows Server 2000. Given that the index.html landing page exists for the server (and its name of WWW), it is highly likely that 10.0.0.133 is the local address of the web server for the company. The data from the packet capture supports this, as two of the biggest percentages of packets going to and from the machine are HTTP and data packets. The other main part of its traffic is DNS, which would make sense as being something that would be needed for other endpoints trying to access a web server.

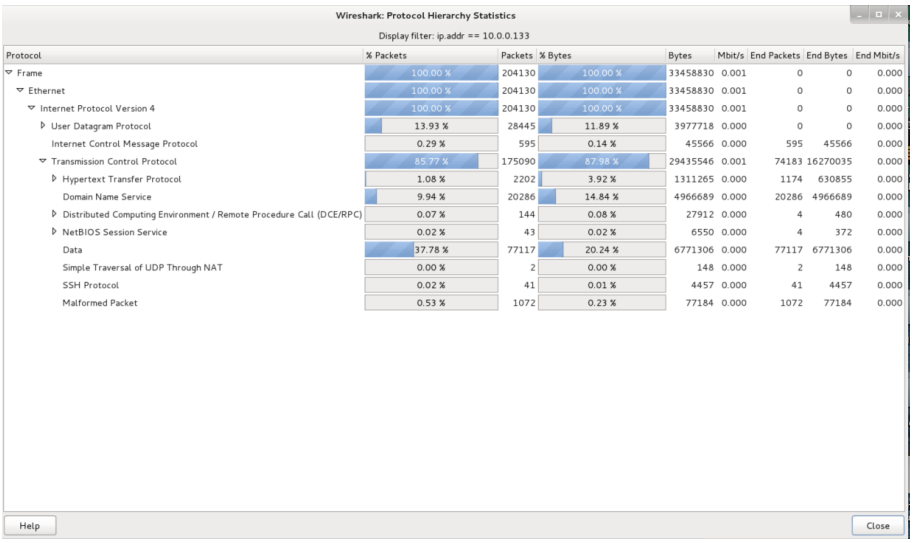


Figure 6: The protocol statistics for packets involving 10.0.0.133 from Wireshark

Through the subsequent parts of my investigation, I found that this machine was the victim machine. This machine was likely configured once as a web server and then generally left there, as there is at least one patch that is seven years old that has still not been applied (more to come on that later.) From its interactions with other machines, I found that this server is running Microsoft IIS, which is generally what Microsoft-based web servers use. It appears that this machine was also configured to be accessed and administered remotely. Using Network Miner, I found that the website places cookies on its users' machines, and also allows for HTTP-POST authentication.

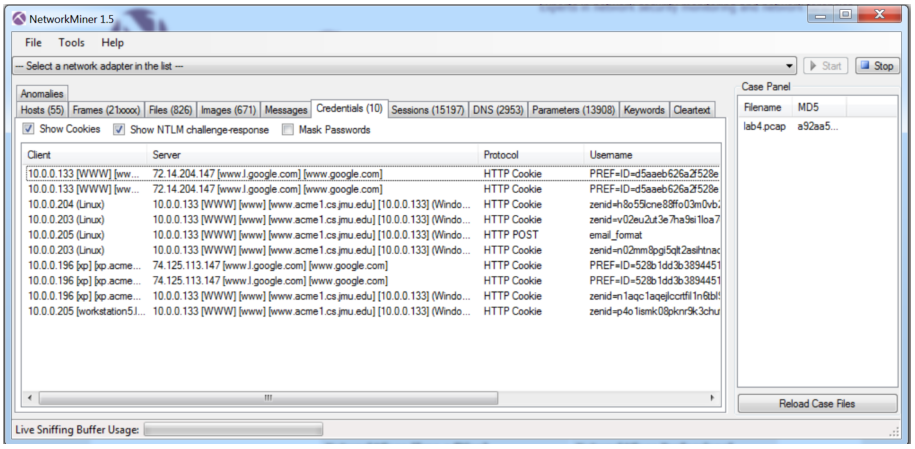


Figure 7: Credentials passed as cookies of HTTP-POST

3.3 10.0.0.196

According to Network Miner, the name of this machine is "xp", and it predicts with a reported 96.83 percent accuracy that the operating system is in fact Windows XP. In addition, the machine makes multiple connections to the Windows Update service at 65.54.81.94, and the machine makes an HTTP request to the web server, with Mozilla reporting its operating system as Windows NT 5.1, which is Windows XP. With all of this evidence, I have a high degree of certainty that this machine is in fact a Windows XP machine.

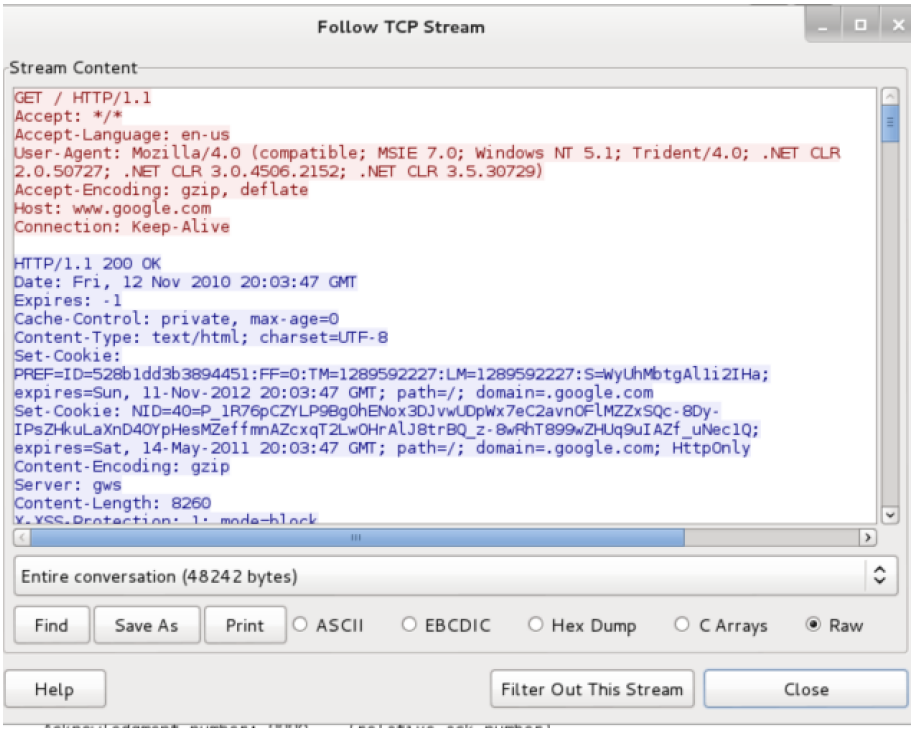


Figure 8: An HTTP session showing that the machine is running XP

While I was investigating this machine, I found evidence of suspicious activity. I will discuss this further in the section entitled "Suspicious and Downright Malicious Activity", but I wanted to note here that I found that this machine was the source of malicious activity on the network.

3.4 10.0.0.201-208

I have grouped together these machines, as they all appear to do the same function. They all have two ports that were used: ports 80 (HTTP) and 22 (SSH). According to Network Miner, each of these machines is running Linux.

In order to verify this, I looked for an HTTP session, and found that Mozilla reported that the machines were running Linux i686. Given that some of these machines have the name workstationX.local, where X is the last digit of its IP (10.0.0.20X), I believe that these machines are all local workstations for doing work on the web server or work relating to the website of the company. The traffic indicates that these machines almost solely connected to the web server (10.0.0.133).

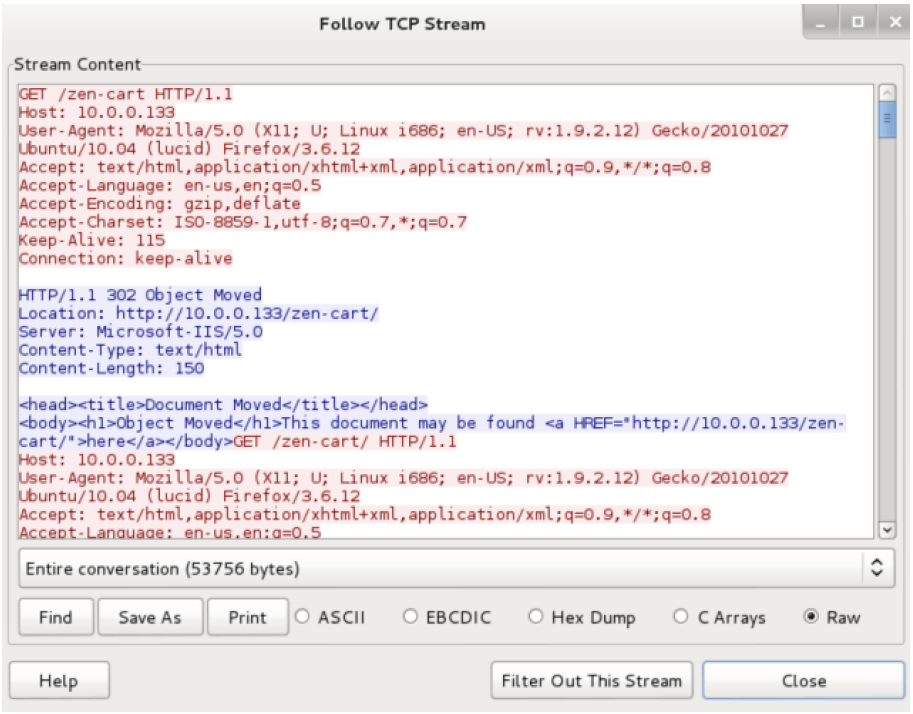


Figure 9: An HTTP session showing that the machine is running Linux

3.5 Overall Network Description

Given all of the above information, here is what I believe the network looks like, with the general flow of information/traffic.

The machine at 10.0.0.1 is the gateway machine for the local network. This machine handles all of the DNS and DHCP for the network. 10.0.0.133 is the address of the public-facing web server. It is running Windows server, which is running Windows IIS. A large amount of the traffic to and from this machine is HTTP traffic. The rest of the machines are endpoints on the network. 10.0.0.201-208 all appear to be configured by the same person, intended to be used interchangably. 10.0.0.196 is the "odd machine out", as it is a Windows XP machine whereas all of the other machines are running Linux. This machine also

runs protocols outside of HTTP and SSH. Given this information, it is possible that this machine was connected to the network without the knowledge of the network administrator, as its configuration is vastly different from the others, but it does not seem to perform any extra tasks that would be relevant to the business.

4 Suspicious and Downright Malicious Activity

A big part of this project was to find out what happened on the network and if any attacks had taken place. Given that I had found that the machine at 10.0.0.196 appeared to be the "odd machine out," I decided to start my invesitgation by looking at what it was doing.

In order to understand what each machine was and what the user tended to do, I followed TCP streams belonging to web traffic for each computer. While doing this, I found that multiple times, computer 10.0.0.196 made many requests to 200.123.107.173. These were typically POST requests to get updates, where the User-Agent was "CORE IMPACT Update Retriever." From previous research, I know that Core Impact is a commercial exploitation tool that normally costs thousands of dollars per year. At the time of writing, Core Impact is listed as the 3rd best explitation tool and the 29th best overall tool on the sectool.org lists of security tools. This was VERY suspicious for the user at 10.0.0196 to have. Because I found this traffic, I decided to take a closer look at how computer 196 interacted with all of the other machines on the network using TCP. I was not as concerned with its use of DNS at this point.

From here, I searched through TCP streams, until I found the one beginning at packet number 80209. This stream shows computer 196 sending a very large amount of data to 10.0.0.133, and eventually getting a windows shell back. From here, I searched backwards to find how the attacker prepared for and executed the attack.

The following is the basic outline of the attack as I interpret it:

1) Recon on the web server

2) Send malformed SMB request to WWW

3) WWW takes a bit of time, and then crashes

4) WWW reboots, re-queries the gateway for DHCP and DNS

5) Send a malformed SMB request to WWW in order to hijack the epmapper pipe

6) Use control of the epmapper to escalate privileges, force WWW to open up port 1566 and have WWW connect back to the attacker's marchine

7) Deliver payload, get a command shell on the WWW server.

The first part of most attacks is reconaissance. I believe to have found multiple instances of scan-like traffic from 10.0.0.196 to the web server. I found evidence that 10.0.0.196 performed UDP scans, RST scans, and SYN scans. The UDP scans and RST scans are straightforward to find, and the evidence of the SYN scan tht I collected is a series of SYN ACKs from the web server to

the attacking machine in short succession. The screenshots of that traffic are below.

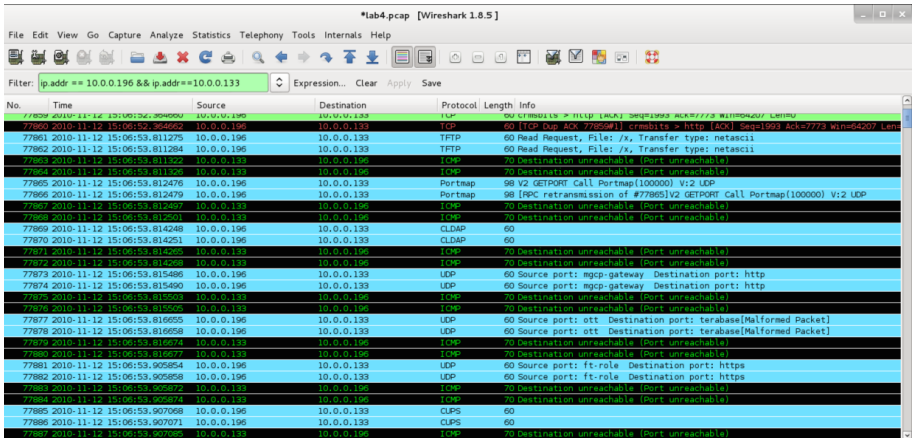


Figure 10: Traffic showing part of the UDP scanning

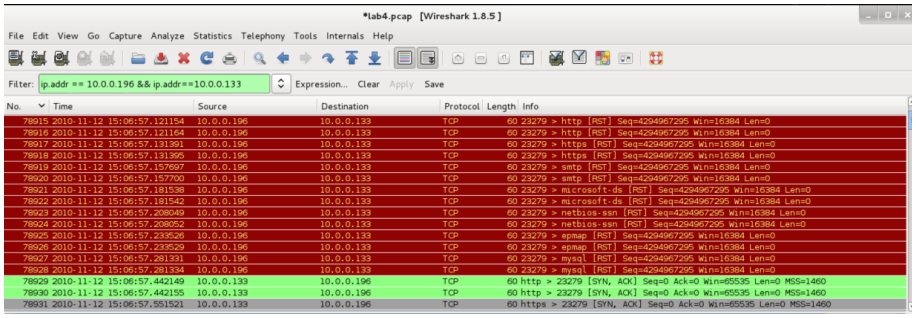


Figure 11: Traffic showing RST scans from the attacker

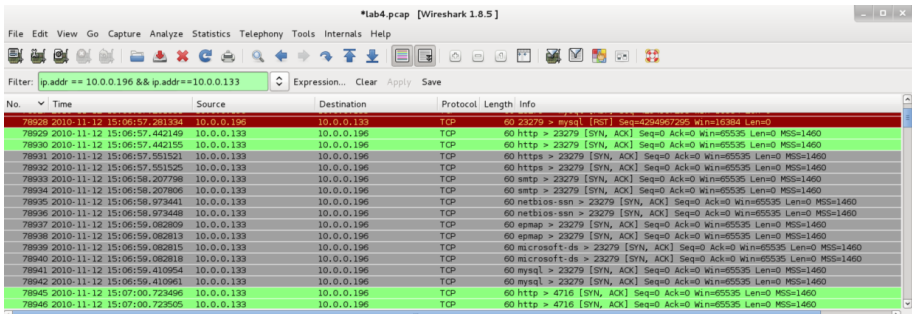


Figure 12: The response from the web server to SYN scanning from the attacker

I searched the CVE database hosted by Mitre, and found one specific vulnerability that this seems to match the rest of the attack exactly: CVE-2003-0605. Mitre's database says this about the vulnerability: "The RPC DCOM interface in Windows 2000 SP3 and SP4 allows remote attackers to cause a denial of service (crash), and local attackers to use the DoS to hijack the epmapper pipe to gain privileges, via certain messages to the RemoteGetClassObject interface that cause a NULL pointer to be passed to the PerformScmStage function." The web page for this entry is: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0605. I found that this exploit is also known as MS03-039 and is discussed in even more detail by Microsoft at:

https://technet.microsoft.com/library/security/ms03-039

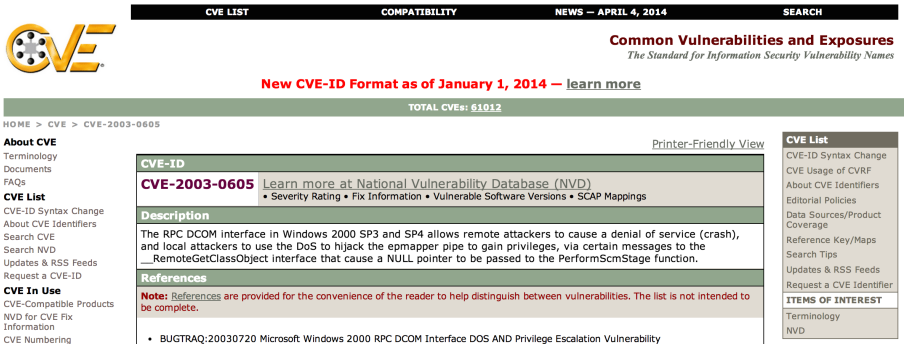
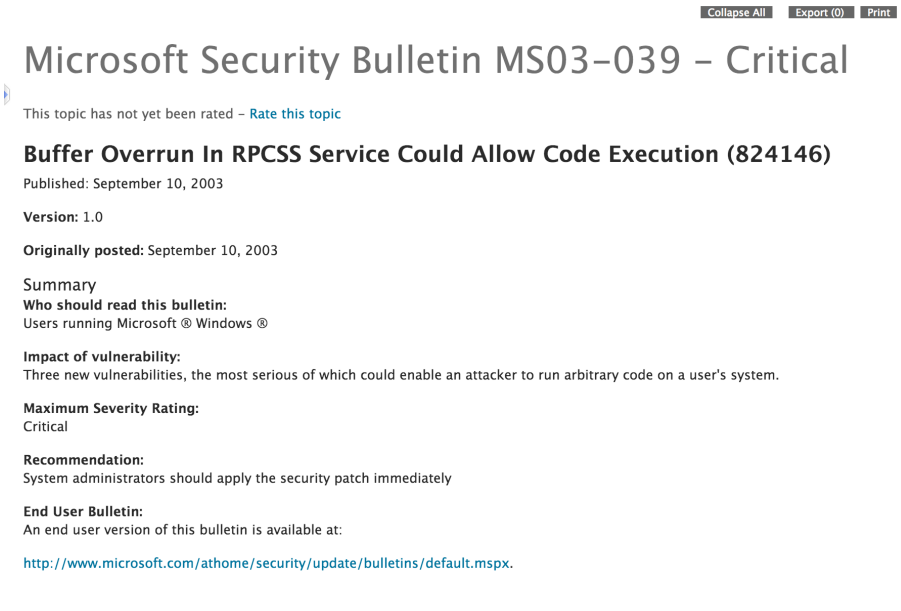


Figure 13: The CVE description from Mitre



Collapse All Export (0) Print

Microsoft Security Bulletin MS03-039 - Critical

This topic has not yet been rated - Rate this topic

Buffer Overrun In RPCSS Service Could Allow Code Execution (824146)

Published: September 10, 2003

Version: 1.0

Originally posted: September 10, 2003

Summary

Who should read this bulletin:
Users running Microsoft ® Windows ®

Impact of vulnerability:
Three new vulnerabilities, the most serious of which could enable an attacker to run arbitrary code on a user's system.

Maximum Severity Rating:
Critical

Recommendation:
System administrators should apply the security patch immediately

End User Bulletin:
An end user version of this bulletin is available at:

http://www.microsoft.com/athome/security/update/bulletins/default.mspx.

Figure 14: The MS Security Bulletin

That seems to be exactly what happened. The web server, as discussed earlier, is in fact a Windows 2000 machine. I found that the attacking machine sent some SMB requests at 15:07:24, and at 15:07:52, only 28 seconds later, the web server is making DHCP and DNS requests that would typically only be made by a computer that just turned on or just connected to the network. The web server was clearly on the network before then, so the most likely explanation is that the web server did in fact crash and reboot.

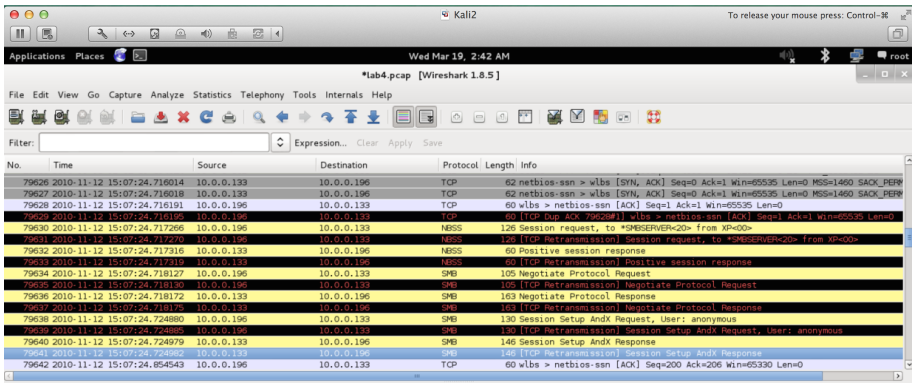


Figure 15: The SMB traffic that caused the crash

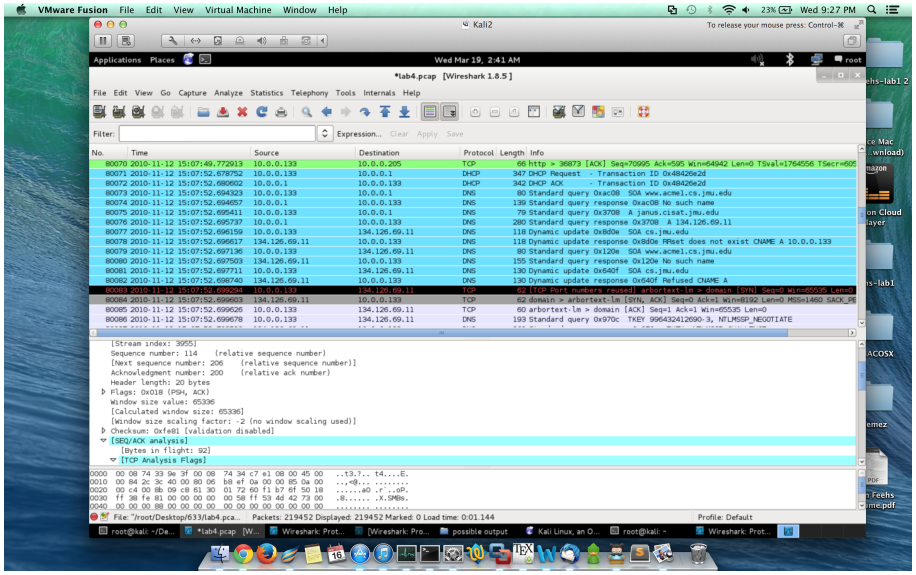


Figure 16: DHCP and DNS requests that show that the web server rebooted

From there, the attack continues to do what the CVE description says it should. The attacker makes a second SMB request to the web server, this time requesting the epmapper pipe (the one the description says can be used maliciously), then calls a RemoteCreateInstance request. After this happens, the web server sends a request to the attacker's machine on a port that was not previously open, and starts a connection that eventually leads to the attacker getting a remote shell on the web server. Below are screenshots showng the SMB traffic attack and the request that shows the web server opening a port for the attacker.

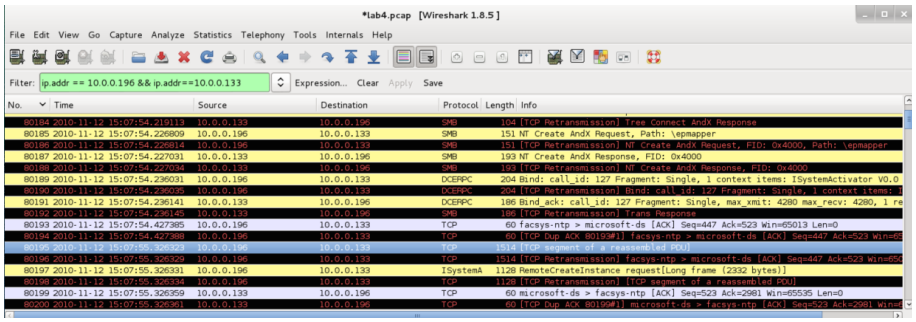


Figure 17: The SMB traffic that the attacker used to exploit the web server

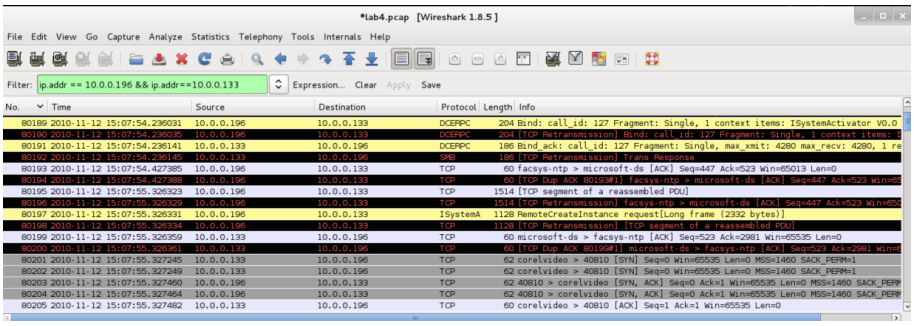


Figure 18: The web server opens up port 1566 and sends a request to the attacker

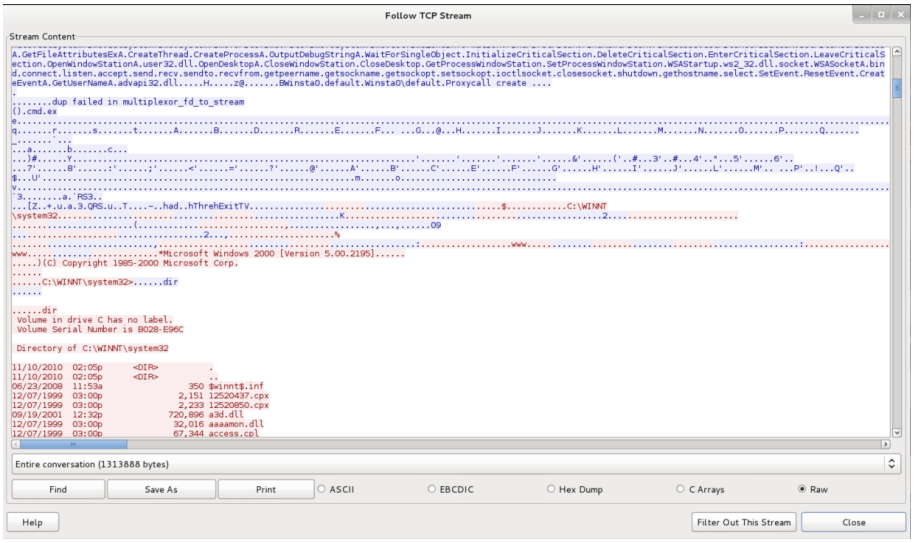


Figure 19: The attacker gets a remote shell on the web server

From here, within the session, the user on computer 196 searches for and reads "secrets.txt" using "more", showing the secret of "banana." The TCP stream then shows the connection eventually ending. This was the end of the traffic that was captured between 10.0.0.196 and 10.0.0.133. In fact, after this, the only thing that the network shoes 10.0.0.196 doing is rebooting/reconnecting to the network, making DHCP and DNS requests to the gateway, updating Core Impact, and then leaving the network. Below are screenshots that show the commands that the attacker used while on the machine, as well as the remainder of 10.0.0.196's actions on the network.

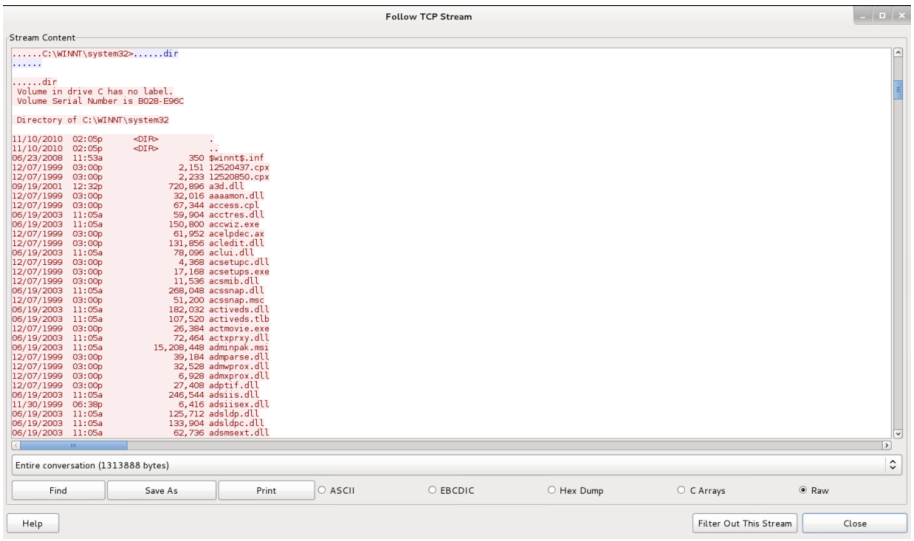


Figure 20: The first command the attacker uses

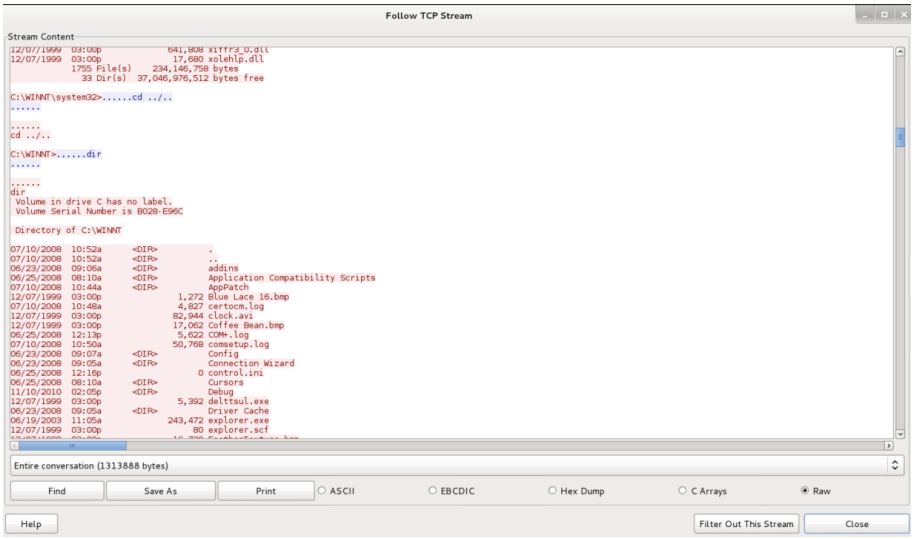


Figure 21: The user continues to probe the web server

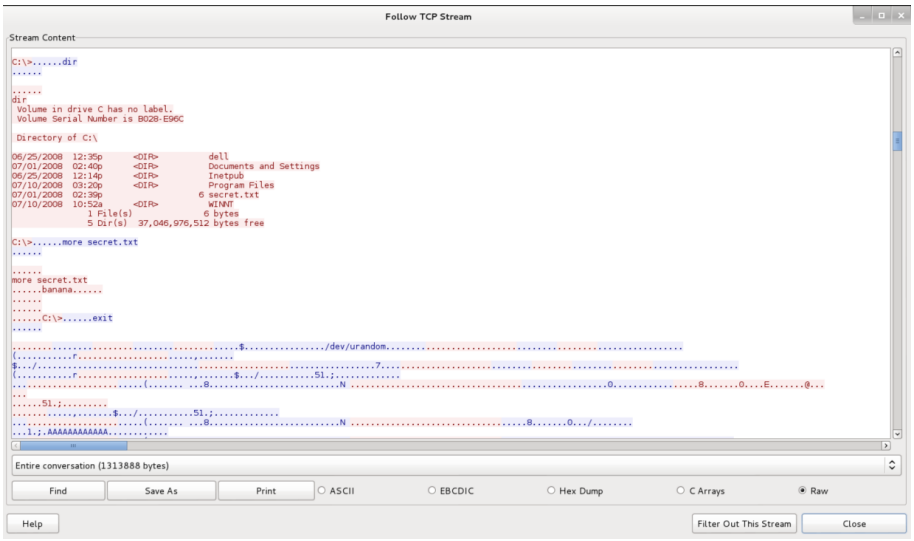


Figure 22: The remainder of the attacker's actions on the web server

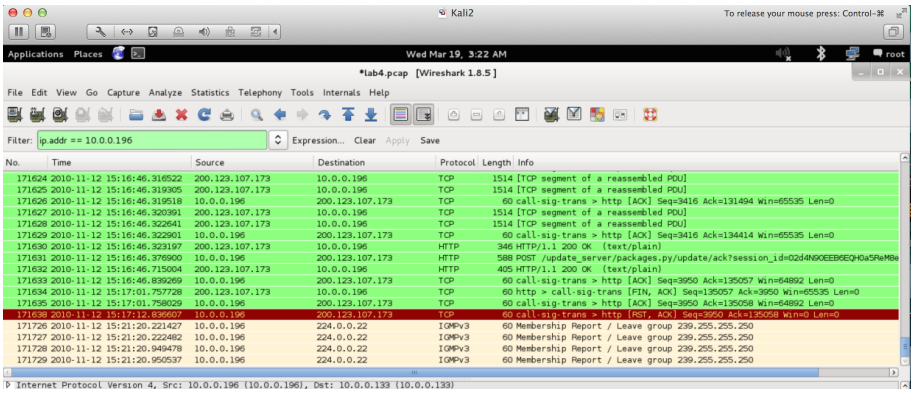
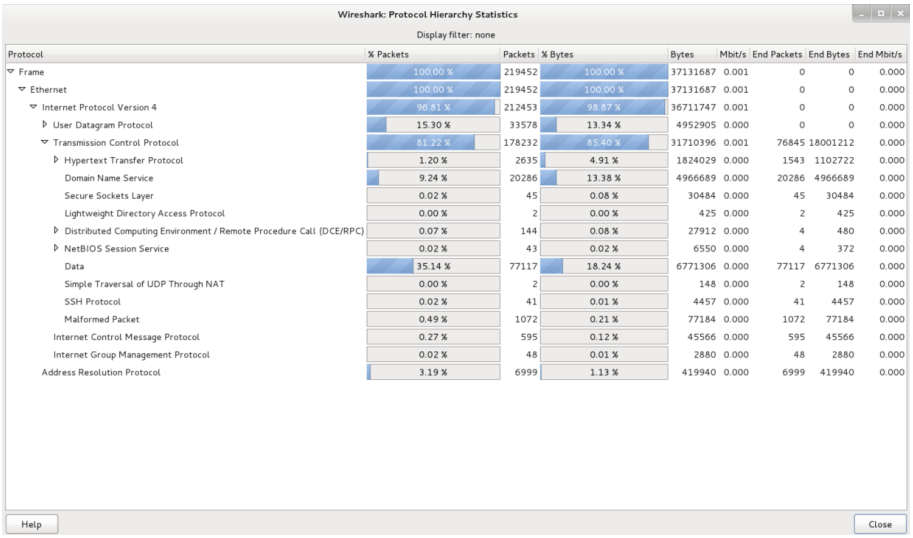


Figure 23: The last traffic captured involving 10.0.0.196

Given that the attacker had Core Impact on their machine and that the attack traffic was relatively large, I believe that this was an automated attack. This is corraborated by the timing of the attack traffic- all of the scan traffic happened in sub-second succession, a frequency that is not likely to be possible for a human to type themselves. Following that line of thought, the speed of the SMB requests and responses is also faster than a human could type. However, it appears that the user still interacted with the system, as there are pauses between each phase of the attack.

5 Non-Attack Actions on the Network (High level)

At a large scale, the network traffic is largely comprised of DNS and HTTP traffic, as well as data sent over HTTP or TCP. I used the statistics section of wireshark to find that those protocols are the most commonly used. Although the screenshot below does not show it, about 70% of the UDP traffic is DNS traffic. I believe that a majority of the non-DNS UDP traffic was generated by the attacker's UDP scanning.



Protocol	% Packets	Packets	% Bytes	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
Frame	100.00 %	219452	100.00 %	37131687	0.001	0	0	0.000
Ethernet	100.00 %	219452	100.00 %	37131687	0.001	0	0	0.000
Internet Protocol Version 4	96.81 %	212453	98.87 %	36711747	0.001	0	0	0.000
User Datagram Protocol	15.30 %	33578	13.34 %	4952905	0.000	0	0	0.000
Transmission Control Protocol	81.22 %	178232	85.40 %	31710396	0.001	76845	18001212	0.000
Hypertext Transfer Protocol	1.20 %	2635	4.91 %	1824029	0.000	1543	1102722	0.000
Domain Name Service	9.24 %	20286	13.38 %	4966689	0.000	20286	4966689	0.000
Secure Sockets Layer	0.02 %	45	0.08 %	30484	0.000	45	30484	0.000
Lightweight Directory Access Protocol	0.00 %	2	0.00 %	425	0.000	2	425	0.000
Distributed Computing Environment / Remote Procedure Call (DCE/RPC)	0.07 %	144	0.08 %	27912	0.000	4	480	0.000
NetBIOS Session Service	0.02 %	43	0.02 %	6550	0.000	4	372	0.000
Data	35.14 %	77117	18.24 %	6771306	0.000	77117	6771306	0.000
Simple Traversal of UDP Through NAT	0.00 %	2	0.00 %	148	0.000	2	148	0.000
SSH Protocol	0.02 %	41	0.01 %	4457	0.000	41	4457	0.000
Malformed Packet	0.49 %	1072	0.21 %	77184	0.000	1072	77184	0.000
Internet Control Message Protocol	0.27 %	595	0.12 %	45566	0.000	595	45566	0.000
Internet Group Management Protocol	0.02 %	48	0.01 %	2880	0.000	48	2880	0.000
Address Resolution Protocol	3.19 %	6999	1.13 %	419940	0.000	6999	419940	0.000

Figure 24: The Wireshark protocol hierarchy

Using network miner, my next goal was to find out what types of HTTP traffic were being generated. Network miner allowed me to see that the sessions that needed credentials belonged either to google.com or to the web server itself.

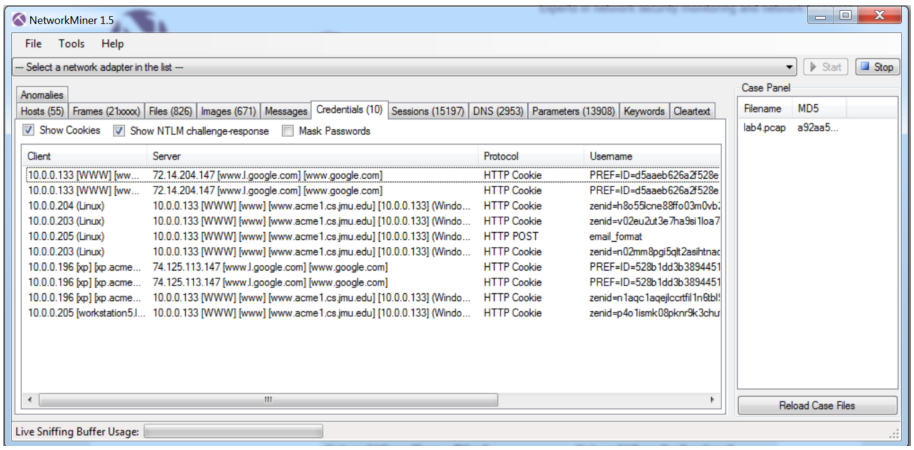


Figure 25: The credentials gathered by Network Miner

From there, I did more digging into what types of information those credentials could give access to. In order to do this, my first step was to use Network Miner to look through the list of files that were returned. A vast majority of these files were web-related files being retreived by the web server. From this, I inferred that a large portion of the data and HTTP packets were web requests to the web server from the workstations, as well as the replies to the workstations from the web server. I also verified this information by looking through the traffic capture in wireshark, and found the same types of HTTP sessions that Network Miner reported existing. Below is a screenshot with data from one of those sessions.

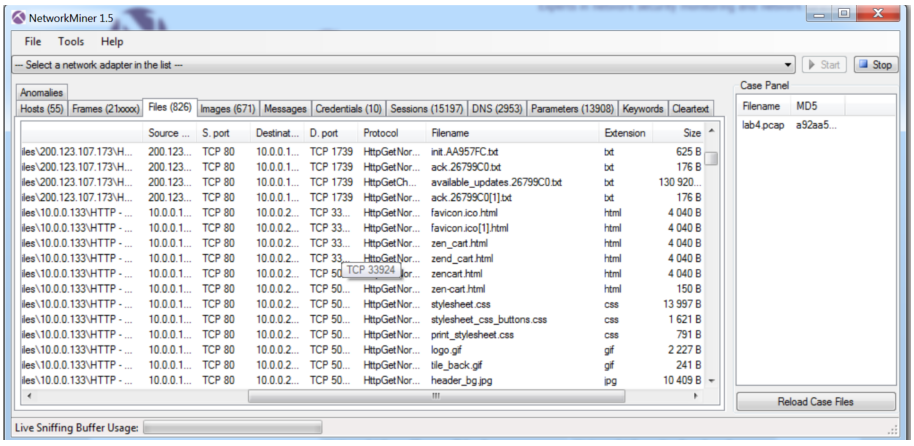


Figure 26: A selection of the files retrieved over the network

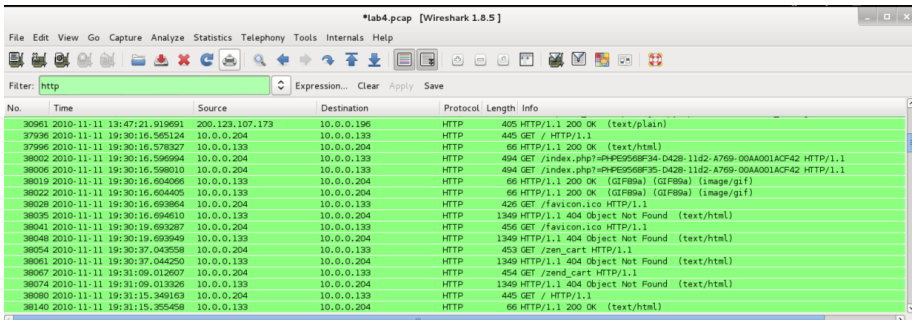


Figure 27: Some of the packets from one HTTP session with the web server and a workstation

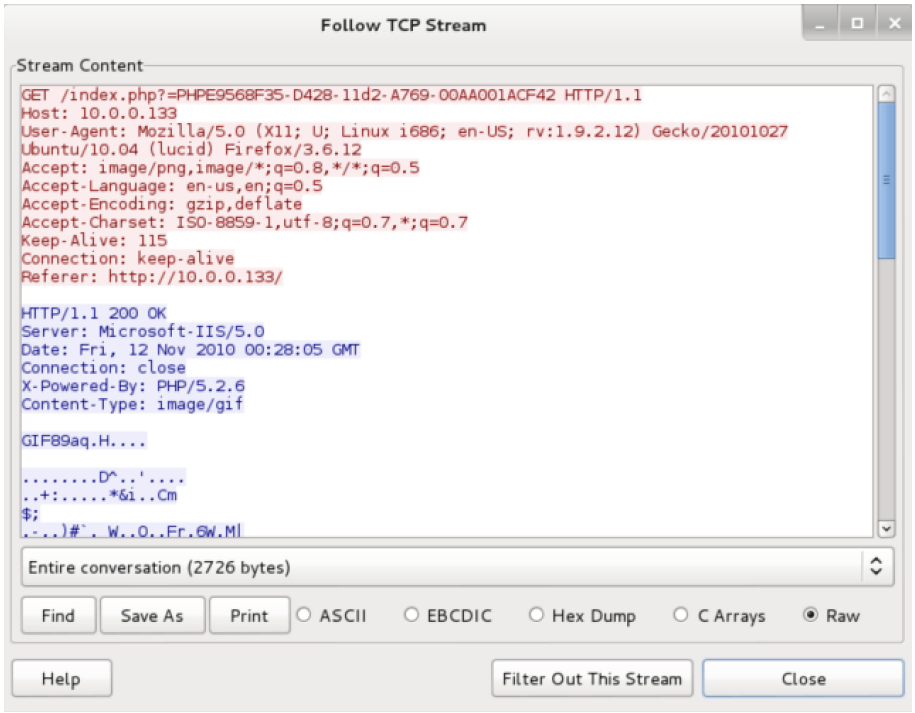


Figure 28: The TCP stream of the above packets

Overall, it appears that the general, non-malicious traffic that was captured is almost all interactions between the workstations and the web server. Some of these use SSH, but a lot of them are actual interactions with the commerce website that involve the part of the site called the "zen-cart," which appears to be the shopping cart for the website. A very large percentage of the traffic is data, as there were 671 images recovered off of the traffic capture. Other traffic

includes windows updates and the occasional query to Google.

This would explain the large amount of DNS traffic as well. As the workstations power on, they need DNS resolution for www.acme1.cs.jmu.edu in order to browse the website. Given the large amount of DNS queries, I believe that the workstations are turned off on a regular basis.

6 Conclusions

In the end, it is clear that the confidentiality of the data in secret.txt on the web serber (10.0.0.133) was breached. In addtion, a malicious user was able to gain remote access to the web server without authorization. Although this traffic capture shows that the attacker did not do much with it, he or she now knows how to get into the web server, and if actions are not taken to stop them from attacking again, they could do even more damage in the future.

With that in mind, I have a few suggestions as to how the attack could have been prevented, and actions to take that will make the entire enterprise more secure. I will discuss these changes from the outside of the network in, and then talk about the changes that could provide the best "bang for the buck" in providing security versus those changes that would be good, but could be costly.

1) Physical security: This is more of a tentative suggestion, as I do not completely know the the company's physical security policies. Given my profile of what endpoints normally look like (see 10.0.0.201-208), there is doubt in my mind that the machine given the IP address 10.0.0.196 should have ever been connected to the network in the first place. It does not fit in with the way the rest of the company's workflow functions, so I believe it is possible that it was not ever supposed to be allowed to connect to the network. If this is the case, and machine 10.0.0.196 is not a company computer, the first way to prevent the attack (or future ones) would be in increase physical security around the network and prevent unauthorized users to connect and also prevent authorized users from connecting unauthorized computers. If 10.0.0.196 had never been able to connect to the network, this attack could not have happened.

2) Network filtering/web audit: While connected to the network, the machine at 10.0.0.196 connected back to the update servers for Core Impact, the tool that was likely used by the attacker to perform the attack. Given that it appears that acme is primarily an e-commerce company (from the evidence in this capture), it is unlikely that any employee would have a good reason to download anything from Core Impact's servers (or Metasploit's, or any other exploitation service.) In order to help prevent this, I would suggest adding website whitelisting at the gateway machine, where only web requests (or other requests) to white-listed domains are allowed. Given the current traffic and trends of the employees, making an initial whitelist should not be too difficult. However, this would introduce extra overhead whenever an employee needs to access a website that is not whitelisted, as they would need to wait for the site to be inspected and then whitelisted before they could access it.

3) Patch management: According to the Microsoft website mentioned earlier,

the vulnerabilty that was exploited by the attacker was published (and patched) in 2003. If this packet capture is from 2010 as Wireshark reports, there was ample time to update the web server and patch this vulnerability. This does not guarantee that the attacker would not have been able to find another way in, but the specific vulnerability that the attacker exploited would not have worked.

4) Encrypt your secrets: The secret.txt file was unencrypted, just sitting on the C drive of the web server. Although its location was not initially evident to the attacker, he or she eventually found it. Security by obscurity adds very little to no value, and had the file been encrypted instead of simply "hidden", even if the attacker still got on the network, called back to Core Impact, and exploited the machine, they still would have needed to crack the encryption on the secret file before being able to access the secrets.

Plausibility: Suggestions 3 and 4 are both fairly easy to do. If the company does not have someone with the technical skills to do both of those right now, there are a lot of good resources online so one of their employees could learn. Suggestion 1 should already be important to any company that does business, stores cart information on their website, and has anything of value. However, depending on the size of the enterprise, increasing physical security to the level that would have prevented this attack may not be feasible. Suggestion 2 would cause the largest increase in work for someone (or multiple people) within the company, as an admin would need to be responsible for the creation and management of the whitelist. Unlike encrypting secrets and only decrypting them when needed, operating a whitelist for web connections could take a significant amount of effort as the enterprise grows, and depending on how acme is doing financially, may not be feasible, even though it could be very helpful.