Analysis of Active and Passive Reconnaissance

COMP 8506 – Assignment 2

Huu Khang Tran, Robert Sobaszek, Anderson Phan, and Peyman Taharni Parsa

Professor: Aman Abdulla

October 2, 2018

Table of Contents

Objective

The purpose of this assignment is to become familiar using a range of tools offered in Kali Linux that ranges from information gathering all the way to penetration testing (Brute Force Attempts or Password Cracking Attempts). We used these tools to get as much information as we can specifically from a targeted host. which meant performing active and passive reconnaissance. In the "Findings" section, we provided a detailed analysis of our findings on the vulnerabilities discovered from using these tools. In the analysis, we will describe what the tool discovered, how the result is justified as a vulnerability, the attacks an attacker can carry out if they were to exploit this, and a defensive strategy to prevent this from happening.

Agreed Partnership for Penetration Testing (Team): Dimitry Rakhlei. Deric, and Prabh
Target Machine IP Address: 24.84.237.85

Tools Used

1. **Zenmap**

   This tool is the graphical user interface for Nmap that allows beginners to discover hosts and services on a computer network (Building a "map" of the intended network) rather than the original command line interface. The main advantage of using Zenmap over Nmap is that It summarizes details about a single host or a complete scan in a convenient display and can also draw a topology map of discovered networks.

2. **SPARTA**

   This tool is a Python GUI application that simplifies network infrastructure penetration testing. SPARTA can be considered a great tool for performing recon/enumeration for which it integrates several reconnaissance techniques into a single and simple GUI. The main advantage of using this tool is that it saves a lot of time by having point-and-click access to toolkits and by displaying all tool output in a convenient way. If little time is spent setting up commands and tools, more time can be spent focusing on analysing results.

3. **Shocker.py (Shell Shock)**

   An open-source Python tool that was installed on the Linux Machine with the main purpose of finding and exploiting webservers that are vulnerable to Shellshock. Shellshock is a

vulnerability founded in Unix Bash shell, also known as a security bug causing Bash to execute commands from environment variables unintentionally. In other words, if exploited the vulnerability allows the attacker to remotely issue commands on the server, also known as remote code execution.

4. **Armitage**

   This tool is a script that collaborates with Metasploit that visualizes targets, recommends exploits, and exposes the advanced post-exploitation features in the framework. Armitage contains tools such as bots that help automate various tasks. It helps to encapsulate, aggregate, and organize the tools found within Metasploit into an interface that's a lot more accessible.

Our Findings

1. FTP Vulnerability using SPARTA

   Figure 1.1 - Using the tool SPARTA to perform a full TCP port scan on machine 24.84.237.85 (Success - Username and Password identified)

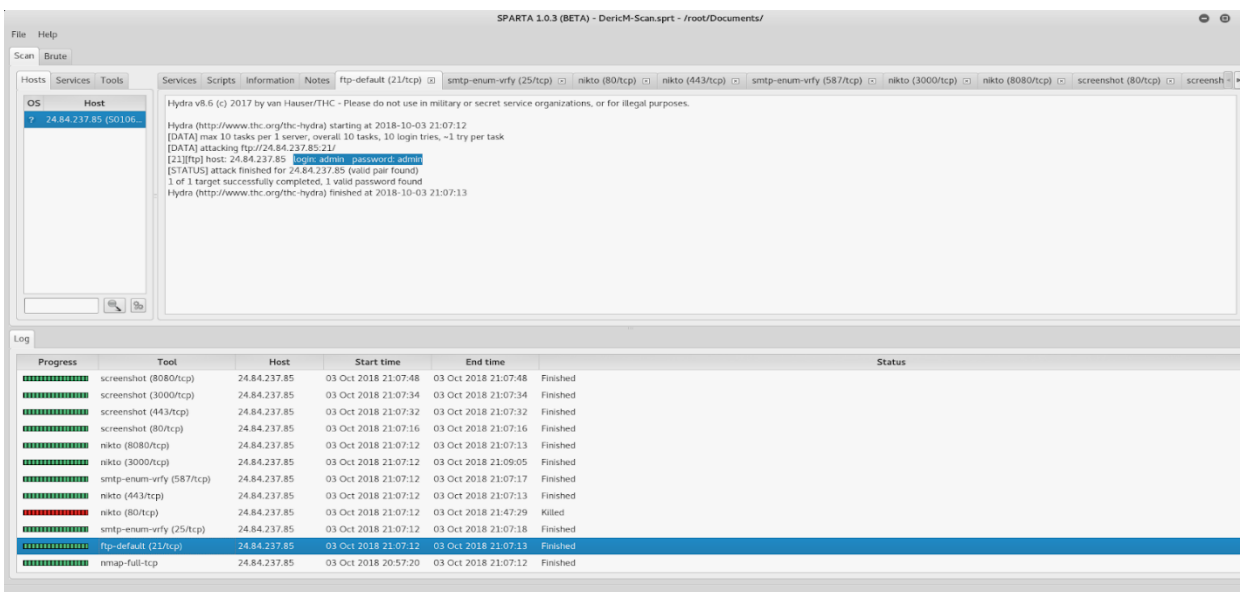
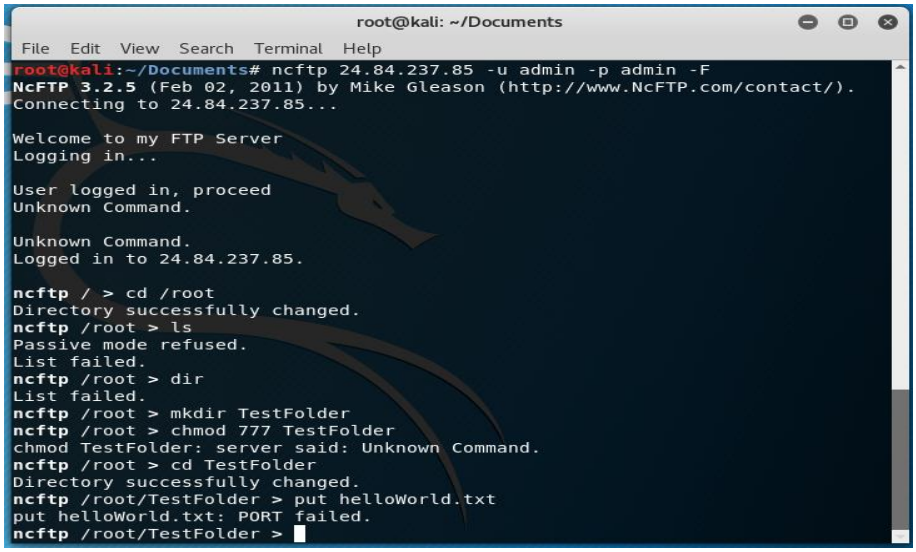


Figure 1.2 - Using the terminal to FTP client our way into target FTP server, showing suspicious results when running general commands



After using the tool, SPARTA, we were able to discover 52 open ports through that could lead to possible vulnerability exploitations like session hijacking, or malware injections. What caught our attention was them having port 21 (FTP) open on their server's end, giving us a way into their filesystem. SPARTA was able to locate a "users file", giving us the required credentials to get into the machine (FTP Server) as shown in Figure 1.1. We then proceeded to use "ncftp 24.84.237.85 -u admin -p admin -E" to act as a ftp client, and managed to get into their file server as shown in Figure 1.2. In an attacker's point of view, once they determine a way to access a hosts FTP server, he/she can exploit this vulnerability by creating hidden backdoors that allows a variety of unauthorized users with root privileges to carry out harmful/malicious agendas. Attacks from unauthorized user can range from theft of critical information (that can be used as a ransomware) all the way to overwriting critical folders that contains key operational elements.

Having an open FTP client session gave us the perfect opportunity to explore the routes an attacker would follow upon exploiting this vulnerability; i.e.. navigating through root directory, changing file/folder permissions, and non-critical file injections. We have reason to believe that this is not a real FTP server based on the results shown in Figure 1.2. To support our justification, figure 1.2 shows general valid commands not being recognized in the environment

("ls", "dir", "put", "jobs" or "umask"), which poses suspicions regarding to not having permission to use these commands. Figure 1.3 also shows that we can change into non-existent directories with no permissions to create/add/remove any files. Summing up these findings leads us to believe that this is a virtual FTP server used to fool attackers in the hopes of identifying these specific users through remote logging.

If we didn't conclude that this was a virtual FTP server and was in fact an active server, a defensive recommendation would be to switch over to SFTP, scrapping the use of FTP as it's old and contains a lack of security perimeters. SFTP is a newer generation of FTP that leverages a more secure connection to transfer files while traversing the filesystem on both the local and remote system. One might want even to add more security to this protocol that can fall in the ranges of having the administrators define specific users allowed in the file system and generate separate digital ssh keys that allows them to connect to the file system remotely without being compromised through middle-man attackers. These are just the defensive recommendations we seek best-fit in addressing FTP vulnerabilities, but there are countless number of ways in enhancing the perimeter security developed by professionals.

2. Shellshock Vulnerability Exploit

Figure 2.1 – The result of scanning the targeted host using Armitage

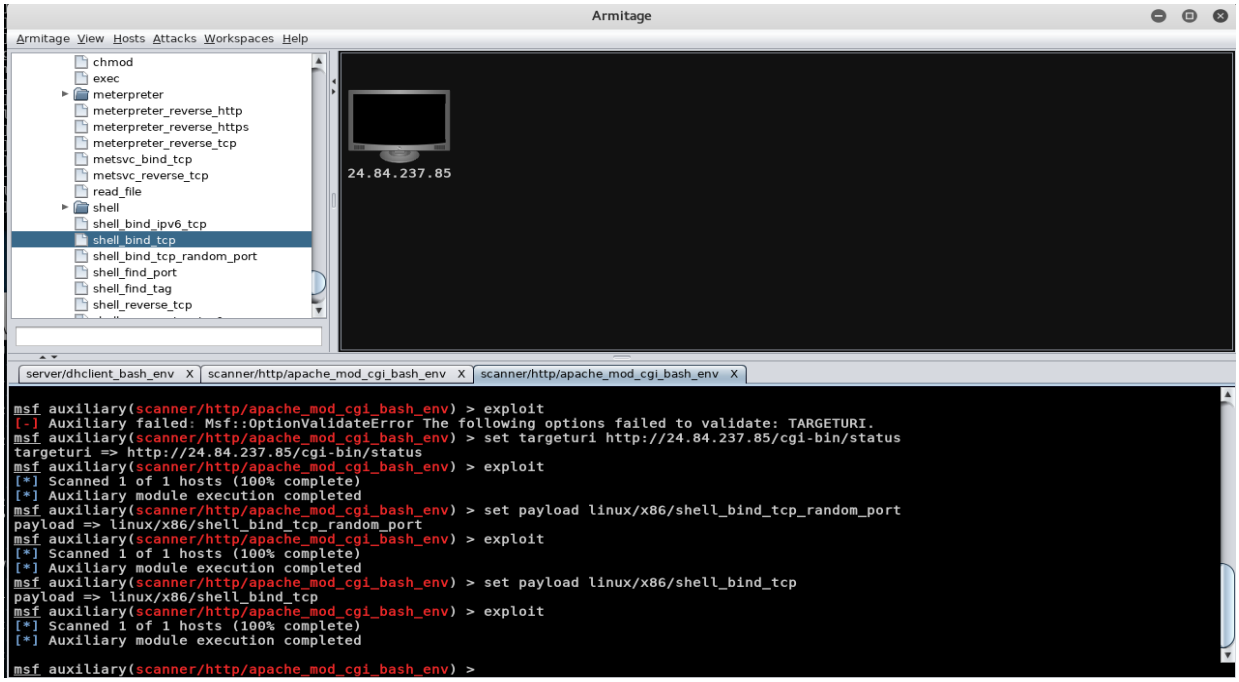
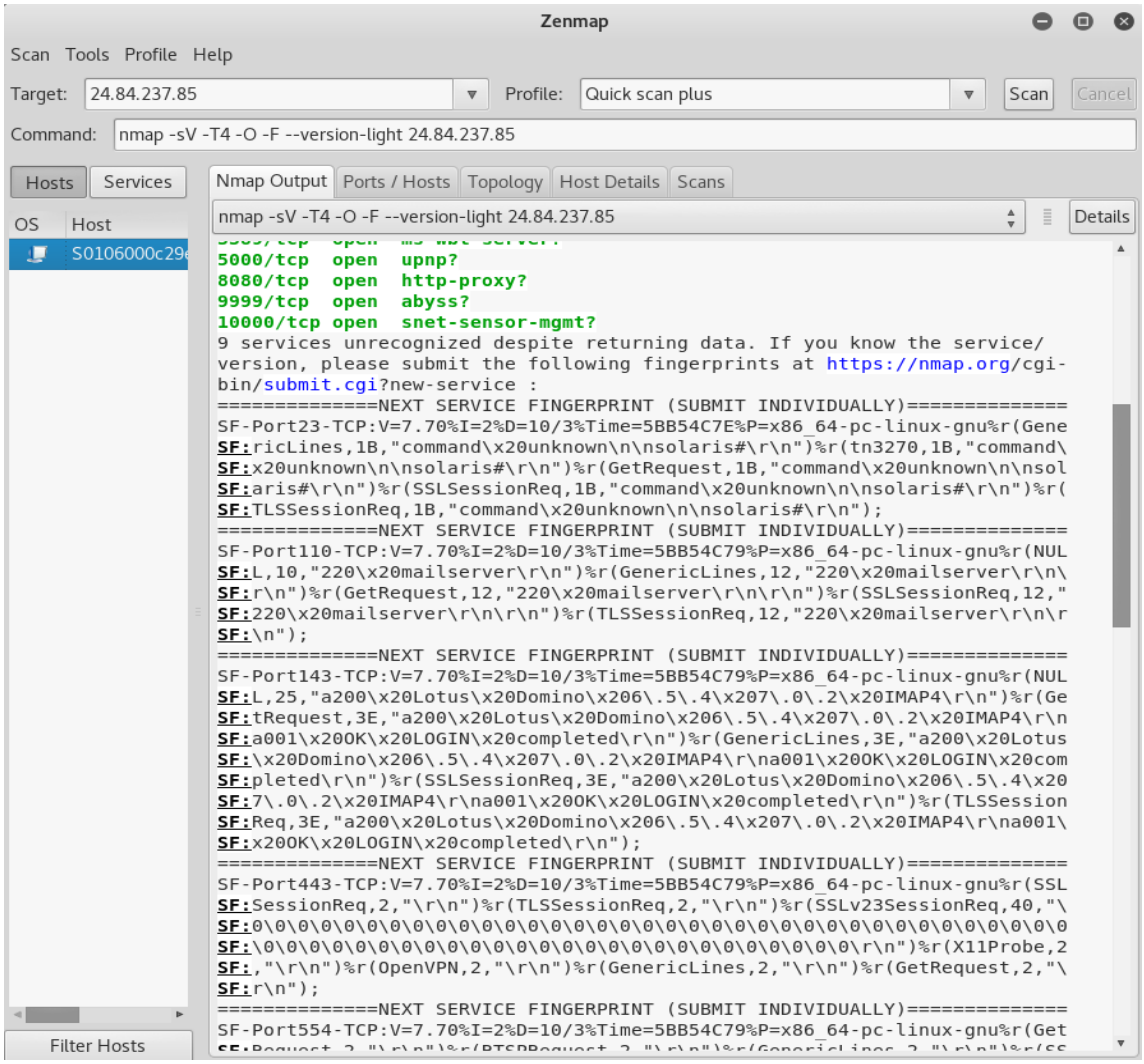


Figure 2.2 – Using Zenmap to display the results after a Nmap scan



Target: 24.84.237.85

Profile: Quick scan plus

Command: nmap -sV -T4 -O -F --version-light 24.84.237.85

```
nmap -sV -T4 -O -F --version-light 24.84.237.85
```

```
5000/tcp  open  upnp?
8080/tcp  open  http-proxy?
9999/tcp  open  abyss?
10000/tcp open  snet-sensor-mgmt?
9 services unrecognized despite returning data. If you know the service/
version, please submit the following fingerprints at https://nmap.org/cgi-
bin/submit.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port23-TCP:V=7.70%I=2%D=10/3%Time=5BB54C7E%P=x86_64-pc-linux-gnu%r(Gene
SF:ricLines,1B,"command\x20unknown\n\nsolaris#\r\n")%r(tn3270,1B,"command\
SF:x20unknown\n\nsolaris#\r\n")%r(GetRequest,1B,"command\x20unknown\n\nsol
SF:aris#\r\n")%r(SSLSessionReq,1B,"command\x20unknown\n\nsolaris#\r\n")%r(
SF:TLSSessionReq,1B,"command\x20unknown\n\nsolaris#\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port110-TCP:V=7.70%I=2%D=10/3%Time=5BB54C79%P=x86_64-pc-linux-gnu%r(NUL
SF:L,10,"220\x20mailserver\r\n")%r(GenericLines,12,"220\x20mailserver\r\n\
SF:r\n")%r(GetRequest,12,"220\x20mailserver\r\n\r\n")%r(SSLSessionReq,12,"
SF:220\x20mailserver\r\n\r\n")%r(TLSSessionReq,12,"220\x20mailserver\r\n\r
SF:\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port143-TCP:V=7.70%I=2%D=10/3%Time=5BB54C79%P=x86_64-pc-linux-gnu%r(NUL
SF:L,25,"a200\x20Lotus\x20Domino\x206\.5\.4\x207\.0\.2\x20IMAP4\r\n")%r(Ge
SF:tRequest,3E,"a200\x20Lotus\x20Domino\x206\.5\.4\x207\.0\.2\x20IMAP4\r\n
SF:a001\x20OK\x20LOGIN\x20completed\r\n")%r(GenericLines,3E,"a200\x20Lotus
SF:\x20Domino\x206\.5\.4\x207\.0\.2\x20IMAP4\r\na001\x20OK\x20LOGIN\x20com
SF:pleted\r\n")%r(SSLSessionReq,3E,"a200\x20Lotus\x20Domino\x206\.5\.4\x20
SF:7\.0\.2\x20IMAP4\r\na001\x20OK\x20LOGIN\x20completed\r\n")%r(TLSSession
SF:Req,3E,"a200\x20Lotus\x20Domino\x206\.5\.4\x207\.0\.2\x20IMAP4\r\na001\
SF:x20OK\x20LOGIN\x20completed\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port443-TCP:V=7.70%I=2%D=10/3%Time=5BB54C79%P=x86_64-pc-linux-gnu%r(SSL
SF:SessionReq,2,"\r\n")%r(TLSSessionReq,2,"\r\n")%r(SSLv23SessionReq,40,"\
SF:0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
SF:\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\r\n")%r(X11Probe,2
SF:,"\r\n")%r(OpenVPN,2,"\r\n")%r(GenericLines,2,"\r\n")%r(GetRequest,2,"\
SF:r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port554-TCP:V=7.70%I=2%D=10/3%Time=5BB54C79%P=x86_64-pc-linux-gnu%r(Get
```

On port 8080, there was an apache web server running, which was notable for investigating as to why it was on a specified port. Exploring that port revealed that the service seemed to be running an older version of bash, which upon look-up revealed that a Shellshock exploit vulnerability was possible to exploit.

The Shellshock exploit that was used on their network was CVE 2014-6271. The way this one works is that it allows an attacker to remotely execute shell commands from malformed environment variables. In this case, it was done on an apache HTTP webserver of their network, then using Armitage the delivery was done with the apache_cgi_bash_env script. Payloads

delivered were scripts that would bind a command shell to a port that listens for connections. (figure 2.1)

Interestingly when running Nmap to do port discovery several ports that were reached would dump out a large amount of information, such as html info and host computer information but garbled up as a service fingerprint. (figure 2.2)

As this exploit can potentially give root access, it is something that must be fixed as soon as possible. Due to the nature of the exploit making use of bash, the only way to really handle this is to update your systems when you can. Detecting this exploit is being used is hard, as the exploit mainly works as a method of delivery of malicious code where the bash just simply executes whatever it sees in the snippet of code.

3. Potential Mail Server Exploit

Figure 3.1 - List of exploits available for Domino from CVE Details

IBM » Lotus Domino » 6.5.4 : Security Vulnerabilities

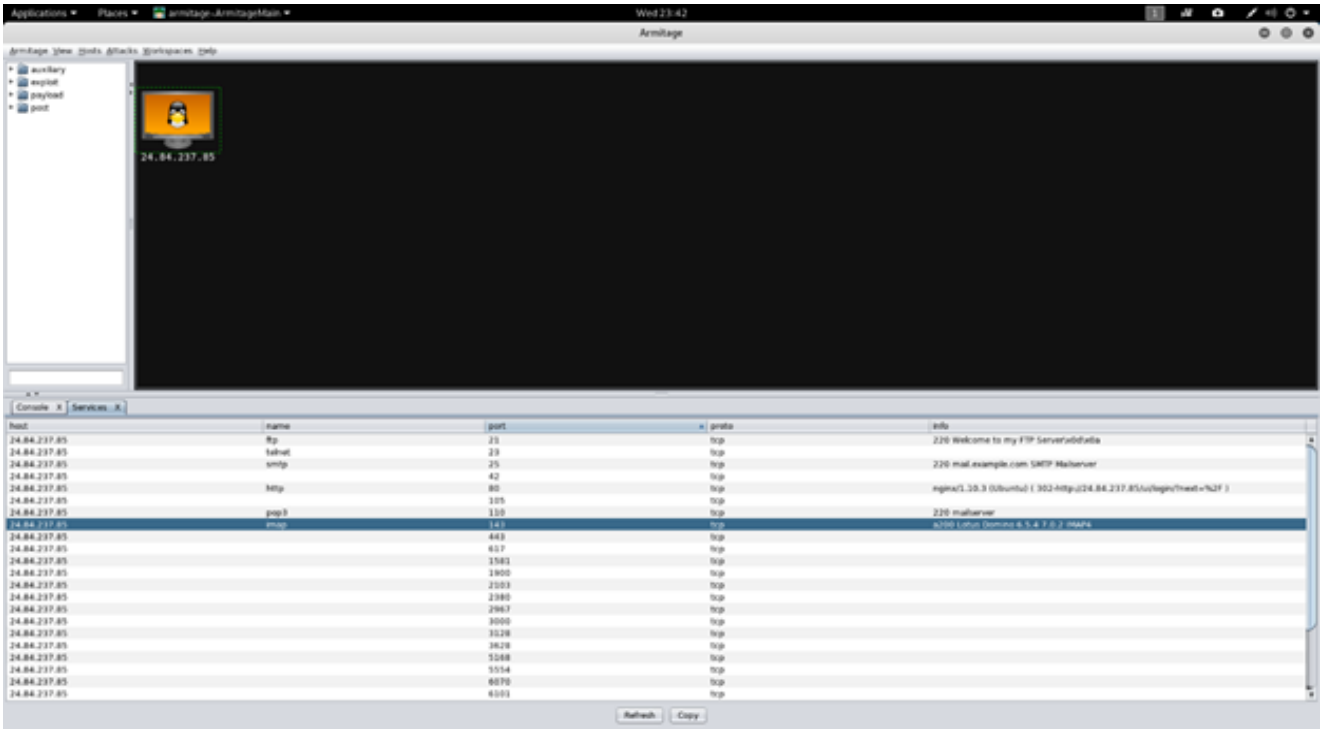
Cpe Name:cpe:/a:ibm:lotus_domino:6.5.4
CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending
Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2011-0915	119		Exec Code Overflow	2011-02-08	2011-04-20	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
Stack-based buffer overflow in nrouter.exe in IBM Lotus Domino before 8.5.3 allows remote attackers to execute arbitrary code via a long name parameter in a Content-Type header in a malformed Notes calendar (aka iCalendar or iCal) meeting request, aka SPR KLYH87LL23.														
2	CVE-2011-0914	189		Exec Code Overflow	2011-02-08	2011-02-23	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
Integer signedness error in ndiiop.exe in the DIIOP implementation in the server in IBM Lotus Domino before 8.5.3 allows remote attackers to execute arbitrary code via a GIOP client request, leading to a heap-based buffer overflow.														
3	CVE-2011-0913	119		Exec Code Overflow	2011-02-08	2011-02-23	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
Stack-based buffer overflow in ndiiop.exe in the DIIOP implementation in the server in IBM Lotus Domino before 8.5.3 allows remote attackers to execute arbitrary code via a GIOP getEnvironmentString request, related to the local variable cache.														
4	CVE-2007-1675			DoS Overflow	2007-03-28	2017-07-28	10.0	Admin	Remote	Low	Not required	Complete	Complete	Complete
Buffer overflow in the CRAM-MD5 authentication mechanism in the IMAP server (nimap.exe) in IBM Lotus Domino before 6.5.6 and 7.x before 7.0.2 FP1 allows remote attackers to cause a denial of service via a long username.														
5	CVE-2006-5818			Exec Code Overflow +Priv	2006-11-08	2017-07-19	7.2	Admin	Local	Low	Not required	Complete	Complete	Complete
Multiple buffer overflows in tunekrnl in IBM Lotus Domino 6.x before 6.5.5 FP2 and 7.x before 7.0.2 allow local users to gain privileges and execute arbitrary code via unspecified vectors.														
6	CVE-2006-4843			XSS Bypass	2007-03-29	2017-07-19	4.3	None	Remote	Medium	Not required	None	Partial	None
Cross-site scripting (XSS) vulnerability in the Active Content Filter feature in IBM Lotus Domino before 6.5.6 and 7.x before 7.0.2 FP1 allows remote attackers to inject arbitrary web script or HTML via unspecified "code sequences" that bypass the protection scheme.														
7	CVE-2006-0121			DoS	2006-01-09	2017-07-19	7.8	None	Remote	Low	Not required	None	None	Complete
Multiple memory leaks in IBM Lotus Notes and Domino Server before 6.5.5 allow attackers to cause a denial of service (memory consumption and crash) via unknown vectors related to (1) unspecified vectors during the SSL handshake (SPR# MKIN67MQVW), (2) the stash file during the SSL handshake (SPR# MKIN693QUT), and possibly other vectors. NOTE: due to insufficient information in the original vendor advisory, it is not clear whether there is an attacker role in other memory leaks that are specified in the advisory.														
8	CVE-2006-0120			DoS	2006-01-09	2017-07-19	5.0	None	Remote	Low	Not required	None	None	Partial
Multiple unspecified vulnerabilities in IBM Lotus Notes and Domino Server before 6.5.5 allow attackers to cause a denial of service (application crash) via multiple vectors, involving (1) a malformed message sent to an "Out Of Office" agent (SPR LPEE6DMQWJ), (2) the compact command (RTINSU2SAJ), (3) malformed bitmap images (MYAA6FHSHW), (4) the "Delete Attachment" action (YPHG6844LD), (5) parsing certificates from a remote Certificate Table (AELE6DZFJW), and (6) creating a SSL key ring with the Domino Administration client (NSUA4FQPTN).														
9	CVE-2006-0119			DoS	2006-01-09	2017-07-19	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
Multiple unspecified vulnerabilities in IBM Lotus Notes and Domino Server before 6.5.5 have unknown impact and attack vectors, due to "potential security issues" as identified by SPR numbers (1) GPKS6C9JE7 in Agents, (2) JGAN6B6TZ3 and (3) KSPR699NBP in the Router, (4) GPKS5YQGPT in Security, or (5) HSAO6BNL6Y in the Web Server. NOTE: vector 3 is related to an issue in NROUTER in IBM Lotus Notes and Domino Server before 6.5.4 FP1, 6.5.5, and 7.0, which allows remote attackers to cause a denial of service (CPU consumption) via a crafted vCal meeting request sent via SMTP (aka SPR# KSPR699NBP).														
10	CVE-2006-0118			DoS Overflow	2006-01-09	2017-07-19	5.0	None	Remote	Low	Not required	None	None	Partial
Unspecified vulnerability in IBM Lotus Notes and Domino Server before 6.5.5, when running on AIX, allows attackers to cause a denial of service (deep recursion leading to stack overflow and crash) via long formulas														

After running a full hail-mary scan in Armitage, we discovered that the host was running Lotus Domino 6.5.4 as an IMAP server (As shown in Figure 3.1). A quick search revealed that this version of Domino has many vulnerabilities that have been exploited. One of the most harmful exploits available is a denial of service attack (CVE-2007-1675) involving login attempts with long usernames. Simply spamming attempts to login with long usernames is enough to have the service buckle and fail. While not damaging to the system, it is both effective, and an extremely obvious attack. There are several other attacks to this version of Domino which allow remote code execution as well, and are far more damaging to the system, while being much more silent. Remote code execution can be achieved by sending a malformed calendar request (CVE-2011-0915). In this way, an attacker could gain complete control of the system with the only potential warning of a corrupt calendar request sent to a user.

The easiest way to mitigate this risk would be to update it to the latest version. However, if this is not possible due to compatibility requirements or some other reason, a firewall rule could be put in place that allows only whitelisted IP addresses to access the service.

Figure 3.2 - Service scan result from Armitage



During our scans, we also noticed that port 22 was open, however we were unable to start an SSH session (As shown in Figure 3.2). This indicates that either interactive logins are disabled, or they are whitelisting IPs that can log in to the machine. We were unable to determine if there was a vulnerability here, but the activity on this port did not seem normal to us.

Conclusion

This assignment gave us a quick glimpse of the tools out there that performs active and passive reconnaissance. These tools simplify the act of penetrating targeted machines through its graphical user interface. Throughout out findings, we were able to fully understand the vulnerability and use our knowledge and experience learned from this class to technically visualize how an attacker can use it against the victim.