COSC362 Assignment

Lewis Garton - 97440534

Alister Marsh - 71624179

Declaration

We declare that the workload contributions to the assignment are as follows:

- Lewis Garton: 50%
- Alister Marsh: 50%

Table of Contents

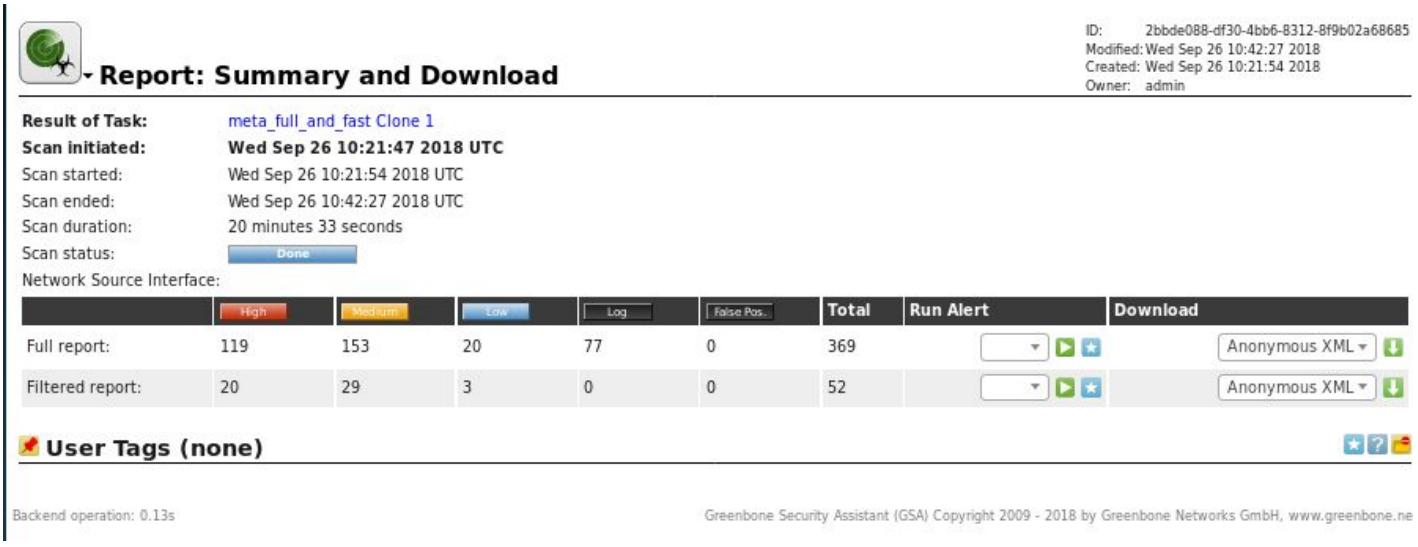
Notes on our setup

- We both ran these exercises on our own systems while in communication with each other. We shared and compared results.
- We used copies of the XP and Metasploitable VMs from the uni servers, but our own copy of the Kali Linux VM. After trying the latest version and finding it glitchy with the Metasploitable and XP machines, we used an older version of Kali which worked.
- We found that after restarting the VMs frequently, the IP addresses would sometimes change, which you may notice in some of the screenshots.

3.1 Vulnerability Scanning (20)

3.1.1: Use OpenVAS for vulnerability scanning (5)

Metasploitable: Report for Full and Fast:



Report: Summary and Download

ID: 2bbde088-df30-4bb6-8312-8f9b02a68685
Modified: Wed Sep 26 10:42:27 2018
Created: Wed Sep 26 10:21:54 2018
Owner: admin

Result of Task:	meta_full_and_fast Clone 1
Scan initiated:	**Wed Sep 26 10:21:47 2018 UTC**
Scan started:	Wed Sep 26 10:21:54 2018 UTC
Scan ended:	Wed Sep 26 10:42:27 2018 UTC
Scan duration:	20 minutes 33 seconds
Scan status:	Done
Network Source Interface:	

	High	Medium	Low	Log	False Pos.	Total	Run Alert	Download
Full report:	119	153	20	77	0	369		Anonymous XML
Filtered report:	20	29	3	0	0	52		Anonymous XML

User Tags (none)

Backend operation: 0.13s

Greenbone Security Assistant (GSA) Copyright 2009 - 2018 by Greenbone Networks GmbH, www.greenbone.ne

Report: Results (52 of 369)

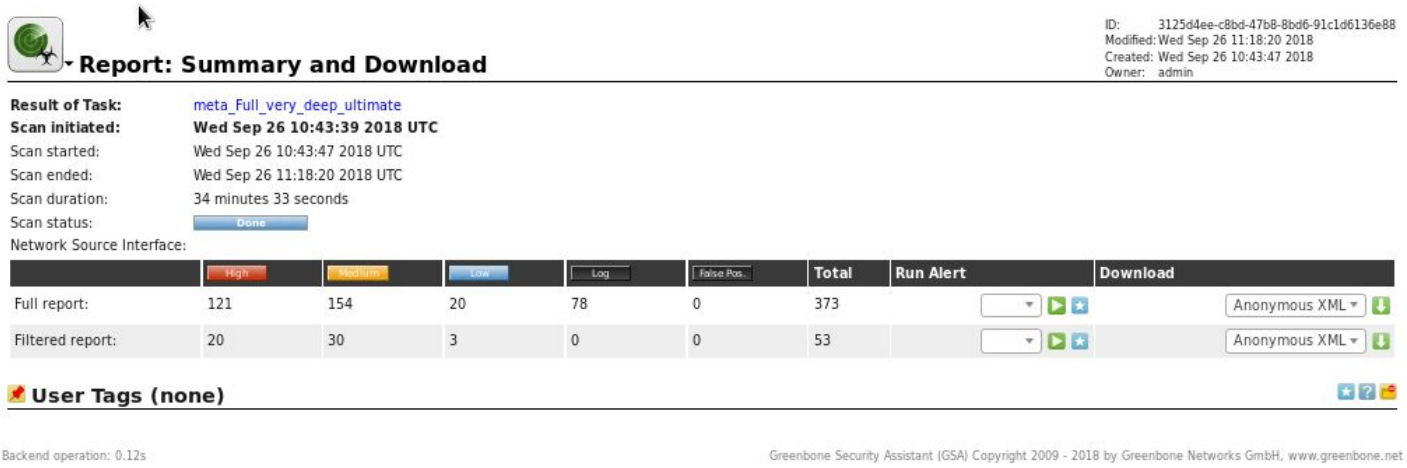
ID: 4bba91ba-129f-4810-8318-35a5e92bf6e7
Modified: Thu Oct 4 06:21:13 2018
Created: Thu Oct 4 06:01:56 2018
Owner: admin

1 - 52 of 52

Vulnerability	Severity	QoD	Host	Location	Actions
Check for rexecd Service	10.0 (High)	80%	192.168.56.103	512/tcp	
OS End Of Life Detection	10.0 (High)	80%	192.168.56.103	general/tcp	
TWiki XSS and Command Execution Vulnerabilities	10.0 (High)	80%	192.168.56.103	80/tcp	
Java RMI Server Insecure Default Configuration Remote Code Execution Vulnerability	10.0 (High)	95%	192.168.56.103	1099/tcp	
Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution Vulnerabilities	10.0 (High)	99%	192.168.56.103	8787/tcp	
Possible Backdoor: Ingreslock	10.0 (High)	99%	192.168.56.103	1524/tcp	
DistCC Remote Code Execution Vulnerability	9.3 (High)	99%	192.168.56.103	3632/tcp	
PostgreSQL weak password	9.0 (High)	99%	192.168.56.103	5432/tcp	
MySQL / MariaDB weak password	9.0 (High)	95%	192.168.56.103	3306/tcp	

The top vulnerabilities were such things as weak passwords, backdoors, and a warning about the OS being out of date. The majority of vulnerabilities had a severity of 0.0, and indicated that the scanner detected services that could be exploitable in certain states or configurations, but not that such a state had been found. An example is a 0.0 severity notification that PostgreSQL is was detected on the target device, which alone is not a problem, but there was an additional 9.0 severity warning that the PostgreSQL password was ‘password’ and thus instantly crackable.

Report for Full and Very Deep Ultimate:



Report: Summary and Download

ID: 3125d4ee-c8bd-47b8-8bd6-91c1d6136e88
Modified: Wed Sep 26 11:18:20 2018
Created: Wed Sep 26 10:43:47 2018
Owner: admin

Result of Task: meta_Full_very_deep_ultimate
Scan initiated: **Wed Sep 26 10:43:39 2018 UTC**
Scan started: Wed Sep 26 10:43:47 2018 UTC
Scan ended: Wed Sep 26 11:18:20 2018 UTC
Scan duration: 34 minutes 33 seconds
Scan status: Done
Network Source Interface:

	High	Medium	Low	Log	False Pos.	Total	Run Alert	Download
Full report:	121	154	20	78	0	373		Anonymous XML
Filtered report:	20	30	3	0	0	53		Anonymous XML

User Tags (none)

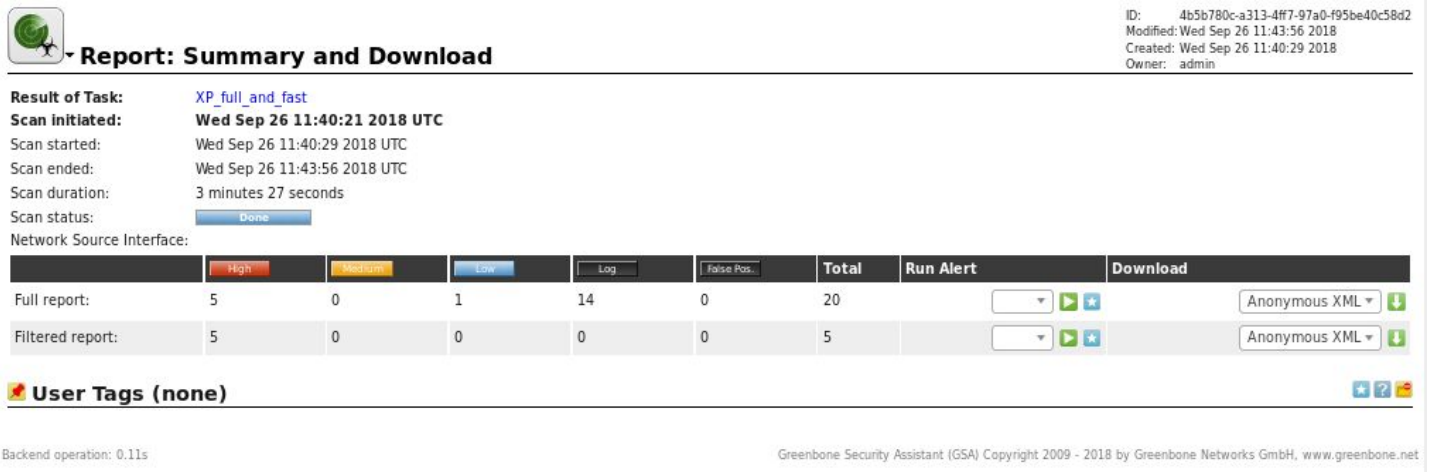
Backend operation: 0.12s

Greenbone Security Assistant (GSA) Copyright 2009 - 2018 by Greenbone Networks GmbH, www.greenbone.net

Our finding was that this mode found one more vulnerability in the ‘medium’ category, and 3 more in the 0.0 severity category. We also note that the scan took around 14 minutes, or 50% longer.

We researched the difference between Full and Fast and Full and Very Deep Ultimate. We discovered that Full and Very Deep Ultimate uses some network vulnerability tests that can potentially interfere with the target machine.

Windows XP report: Full and Fast



Report: Summary and Download

ID: 4b5b780c-a313-4ff7-97a0-f95be40c58d2
Modified: Wed Sep 26 11:43:56 2018
Created: Wed Sep 26 11:40:29 2018
Owner: admin

Result of Task: XP_full_and_fast
Scan initiated: **Wed Sep 26 11:40:21 2018 UTC**
Scan started: Wed Sep 26 11:40:29 2018 UTC
Scan ended: Wed Sep 26 11:43:56 2018 UTC
Scan duration: 3 minutes 27 seconds
Scan status: Done
Network Source Interface:

	High	Medium	Low	Log	False Pos.	Total	Run Alert	Download
Full report:	5	0	1	14	0	20		Anonymous XML
Filtered report:	5	0	0	0	0	5		Anonymous XML

User Tags (none)

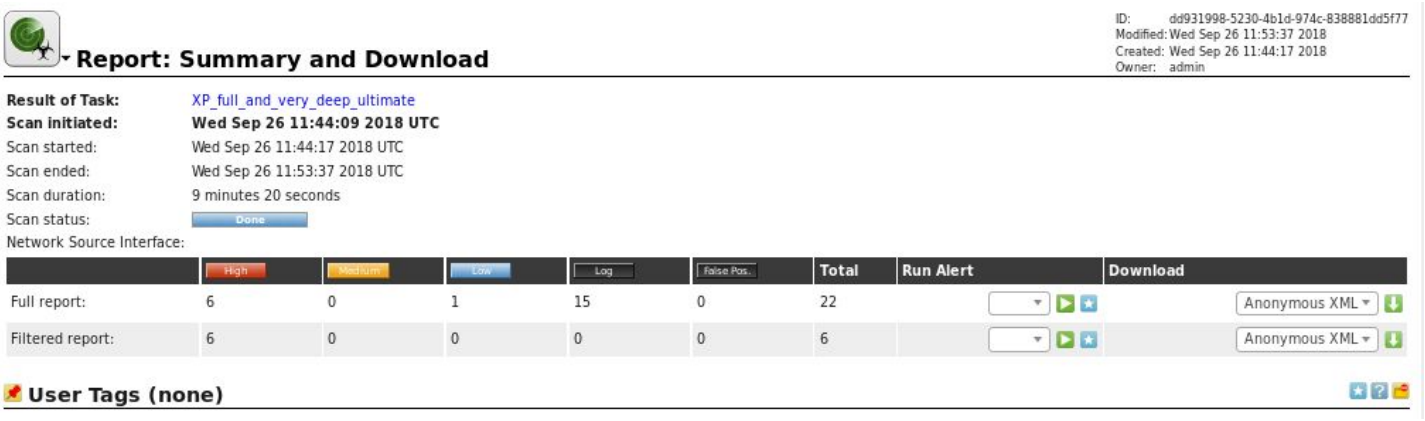
Backend operation: 0.11s

Greenbone Security Assistant (GSA) Copyright 2009 - 2018 by Greenbone Networks GmbH, www.greenbone.net

We initially had the problem that OpenVAS was only finding one vulnerability (the OS End of Life check). The reason for this was unknown. Lewis deleted the virtual machines and reinstalled them and we detected 5 high priority vulnerabilities.

One observation is that the scan only took 3.5 minutes, which is about six times faster than the Metasploitable scan. This could be due to XP not running as many exploitable services as Metasploitable.

Windows XP report: Full and Very Deep Ultimate



Report: Summary and Download

ID: dd931998-5230-4b1d-974c-838881dd5f77
Modified: Wed Sep 26 11:53:37 2018
Created: Wed Sep 26 11:44:17 2018
Owner: admin

Result of Task: XP_full_and_very_deep_ultimate
Scan initiated: **Wed Sep 26 11:44:09 2018 UTC**
Scan started: Wed Sep 26 11:44:17 2018 UTC
Scan ended: Wed Sep 26 11:53:37 2018 UTC
Scan duration: 9 minutes 20 seconds
Scan status: Done
Network Source Interface:

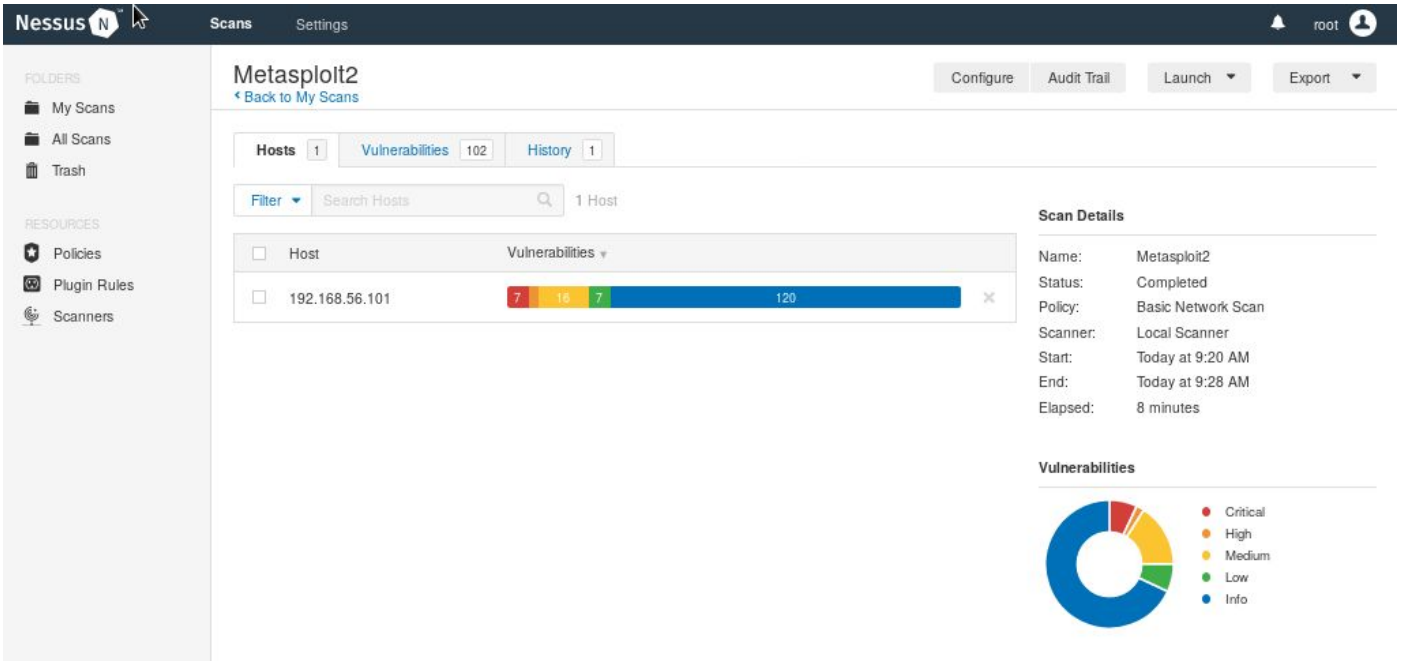
	High	Medium	Low	Log	False Pos.	Total	Run Alert	Download
Full report:	6	0	1	15	0	22		Anonymous XML
Filtered report:	6	0	0	0	0	6		Anonymous XML

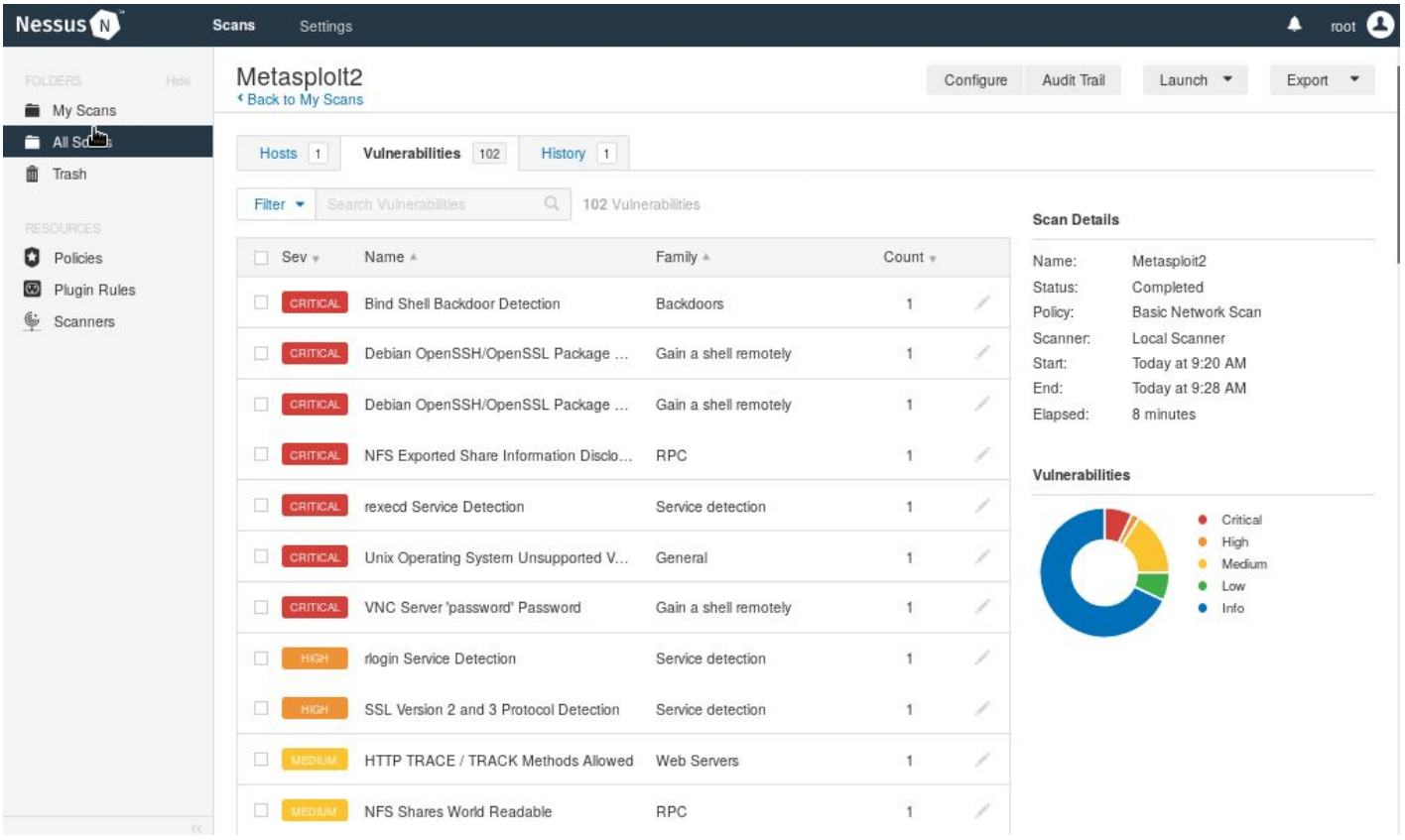
User Tags (none)

We found that this mode detected one more vulnerability (ms08_067 NetAPI32.dll server service), which is quite a reliable vulnerability to exploit. It took 6 minutes, or about 200% longer than the Full and Fast scan above.

3.1.2: Use Nessus for vulnerability scanning (5)

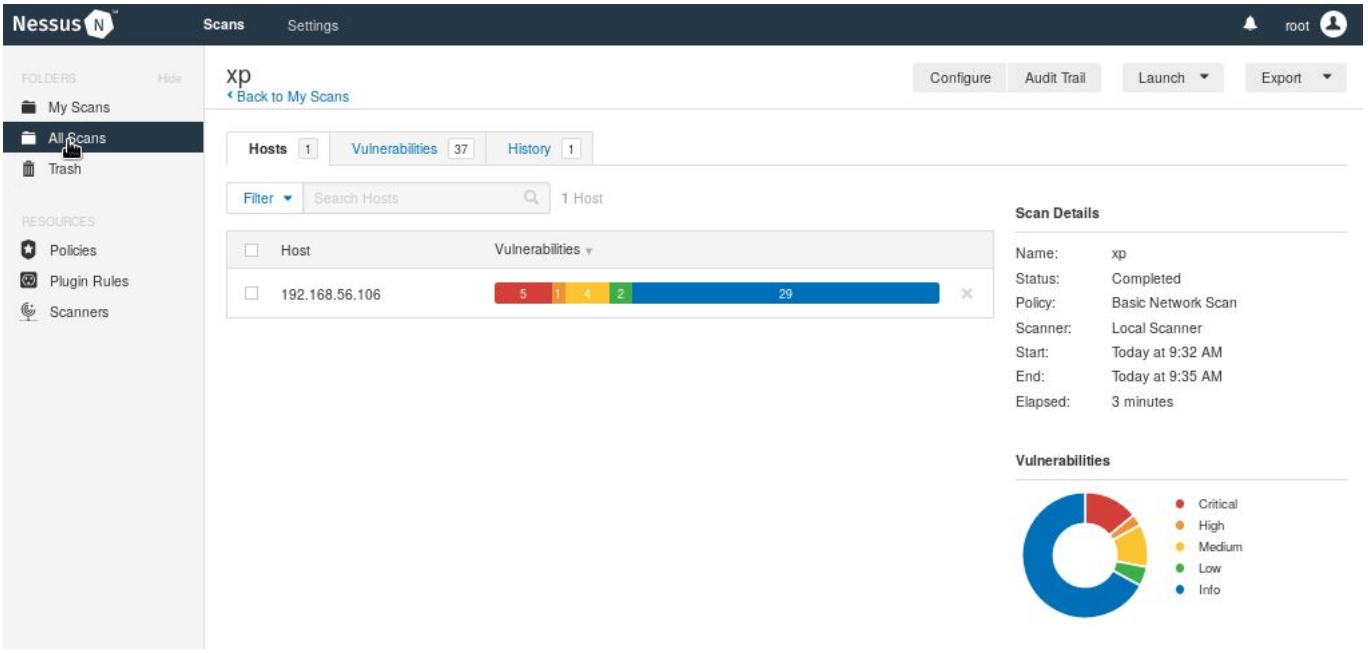
Metasploitable Report:



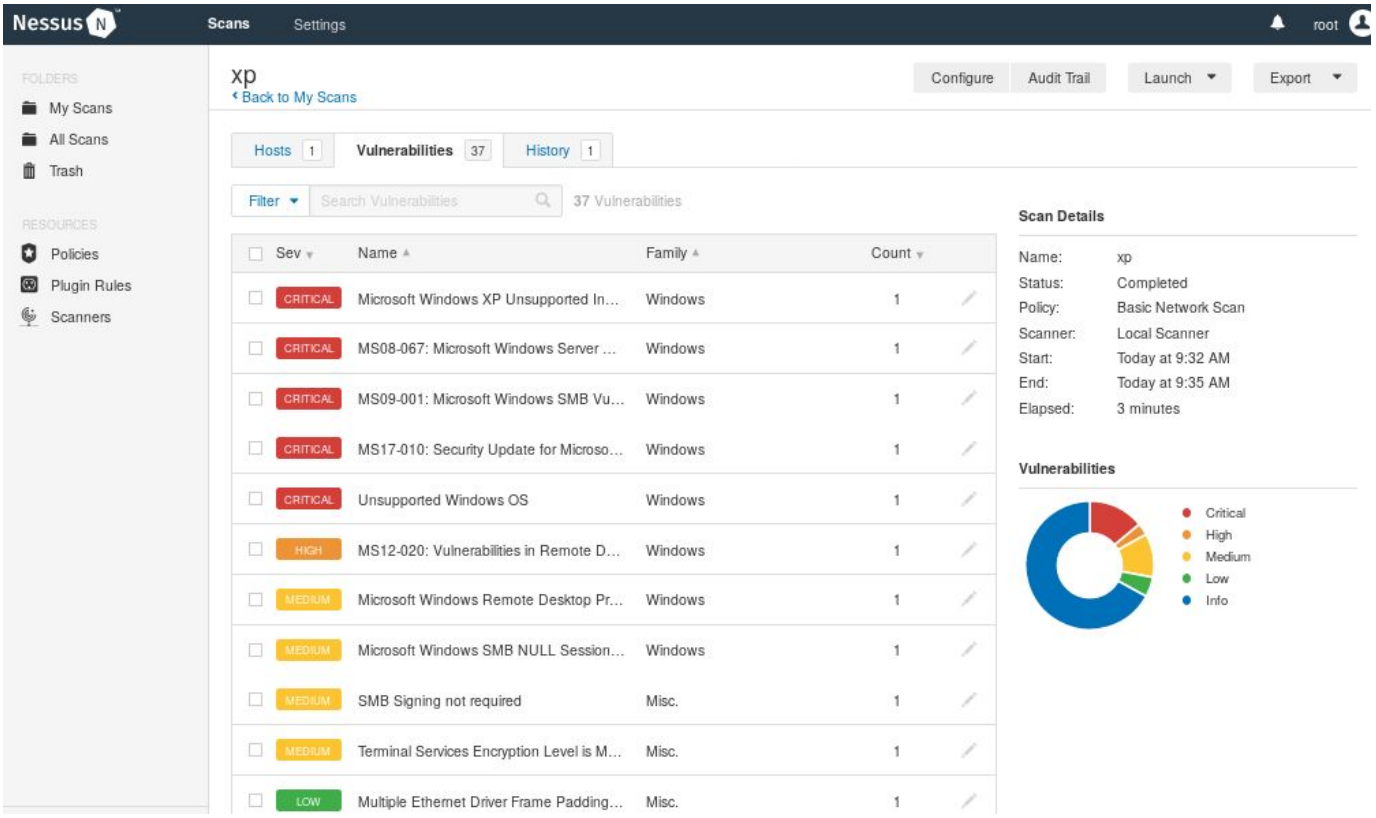


We initially had problems with Nessus not returning any vulnerabilities. Switched to NAT to update Nessus and install plugins, which fixed the problem. The scan took 8 minutes, which was faster than the OpenVAS scan. It did, however, report fewer vulnerabilities.

Windows XP Scan Report:



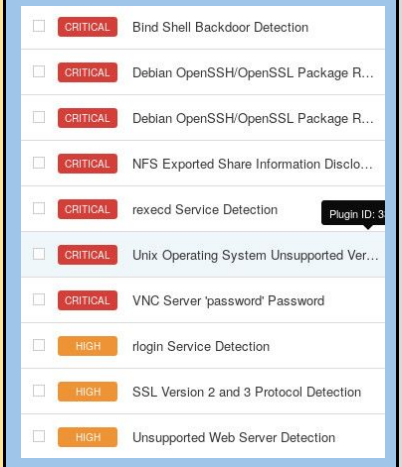
Nessus identified 12 vulnerabilities of low or higher severity. Lewis and Alister obtained identical results (other than Alister's scan completing more quickly). For the comparisons in the next question, we are using the times taken on Lewis' system. This was more vulnerabilities than the OpenVAS scanner found, and like the OpenVAS scanner Nessus took less time to scan the XP machine than the Metasploitable machine.



3.1.3: Compare OpenVAS and Nessus (10)

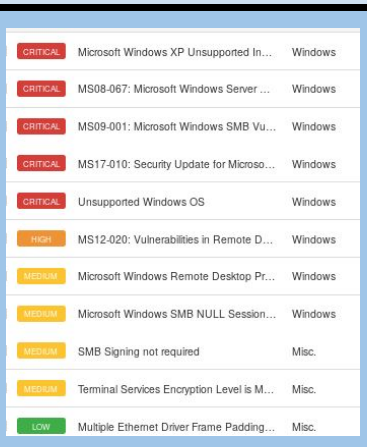
We both ran the scans individually and compared results. As we were running copies of the same virtual machines, we were not surprised that the results were identical in all cases.

Metasploit Scan Results Comparison:

	OpenVAS	Nessus	Analysis
Number of Vulnerabilities	**52** found. 20 high 29 medium 3 low. (This is not including severity 0.0 vulnerabilities, which were filtered out).	**34** found. 7 critical. 3 High. 17 medium. 7 Low. (This does not include 'info' level vulnerabilities.)	OpenVAS found a higher total of vulnerabilities that it considers to be a risk (eg. of low or higher severity). It also reported about twice the number of 'high' severity vulnerabilities.
Top Vulnerabilities	 *Note: This includes all vulnerabilities of severity 10.0 and some of 9.*	 *Note: This is all vulnerabilities of **critical** or **high** severity.*	Top vulnerabilities are somewhat different between the two. Some vulnerabilities were reported by both but were named quite differently (see below). OpenVAS has reported a number of vulnerabilities related to weak passwords, while of these Nessus only reported VNC weak password. Both reported `rexecd` as one of the top vulnerabilities. 'Bind shell backdoor detection' and 'possible backdoor: ingreslock' may refer to the same vulnerability. Also, 'OS End of Life Detection' and 'Unix Operating System Unsupported Version' appear to refer to the same vulnerability. There are probably other matches as well.
Time Taken	**20** min **33** s on Full and Fast **34** min **33** s on Full and Very Deep Ultimate	**8** minutes	Nessus completed much more quickly.
Report detail	Vulnerabilities ranked on a continuous 10 point	Vulnerabilities classified by category (critical to low).	We found OpenVAS hard to navigate. It took longer, but was

	scale. Get to list of threats found by clicking on the number of threats. Report combines all vulnerabilities into one category.		more detailed.
Summary	Overall, OpenVAS found (or claimed to find) more vulnerabilities than Nessus. It spent more time scanning. It found more vulnerabilities if the Very Deep Ultimate mode was used.		

Windows XP Scan Results Comparison:

	OpenVAS	Nessus	Analysis
Number of Vulnerabilities	**7** found 6 high 1 low	**12** found. 5 critical 1 high 4 medium 2 low	Nessus appears to have found more vulnerabilities.
Top Vulnerabilities			Both scanners identified the non-supported OS as being a top vulnerability.
Time Taken	**3** minutes **27** s on Full and Fast **9** minutes **27** on Very Deep Ultimate	**3** minutes (Lewis' system)	Nessus was faster, particularly compared to OpenVAS's Very Deep Ultimate mode, which took three times as long.
Report detail	As above	As above	As above
Summary	For the XP scans, Nessus appeared to find more vulnerabilities in less time. OpenVAS's Very Deep Ultimate scan took considerably more time than its fast mode.		

Overall Comparison:

Our general takeaways regarding the scanners are as follows:

- Both scanners found different vulnerabilities. Neither (and probably no) scanner will find all vulnerabilities.
- Changing the type of the scan on OpenVAS can definitely find more vulnerabilities. When we did so we found an additional high priority vulnerability for Windows XP.
- One scanner seemed to perform better at scanning Metasploitable (OpenVAS) and the other at scanning XP (Nessus). This would suggest that trying different scanners on any given OS would be a good idea.
- We found the Nessus reports much easier to read than the OpenVAS reports.
- Lewis found OpenVAS crashed a lot for him before completing a successful scan, but Nessus did not.

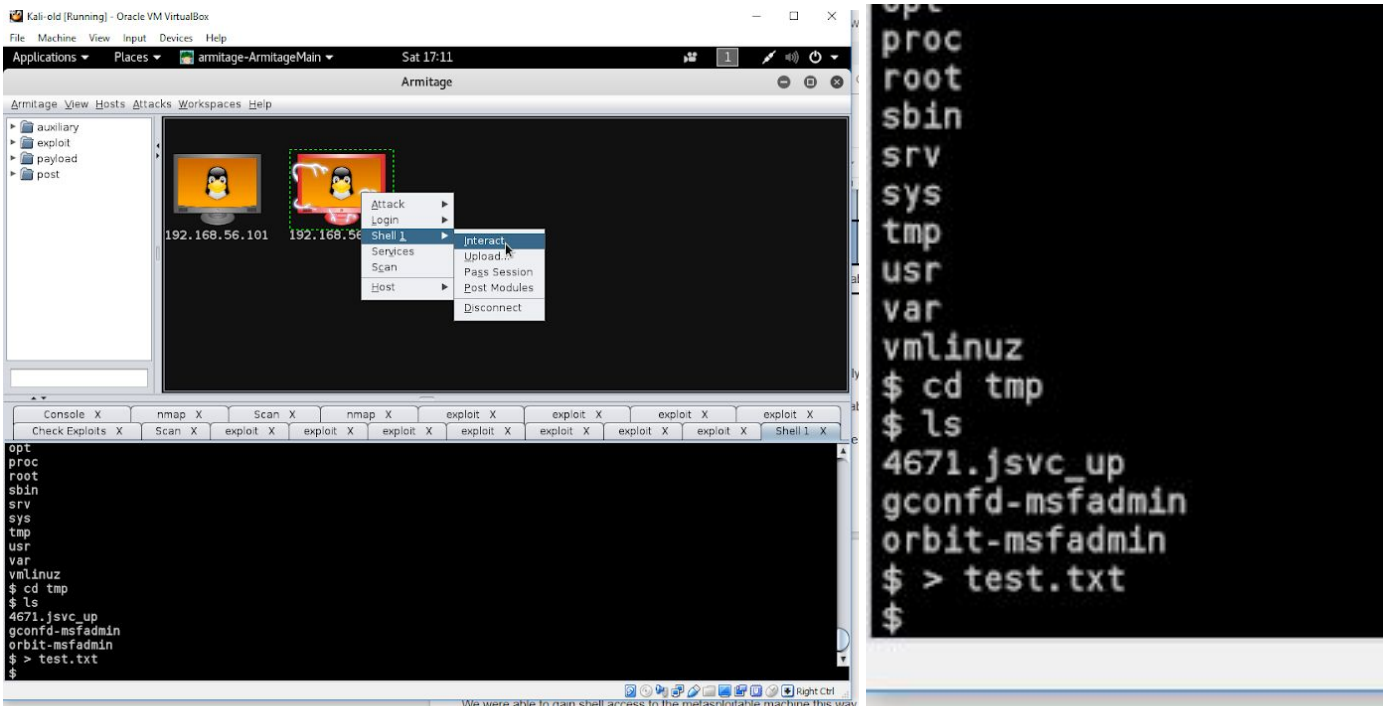
3.2 Exploitation and Exfiltration (40)

3.2.3 Exploiting Metasploitable (10)

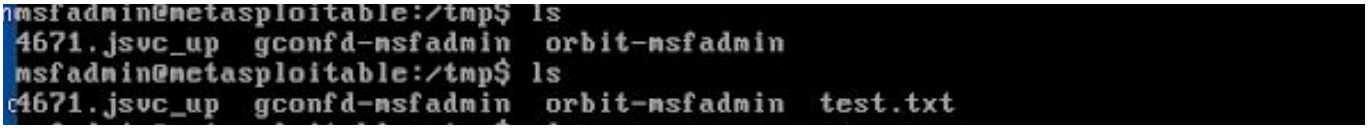
We gained root access to the metasploitable machine in the following ways:

Unreal_IRCD_3281 Backdoor Exploit

We were able to gain root access to the metasploitable machine with the Unreal IRC daemon backdoor https://en.wikipedia.org/wiki/UnrealIRCd. We confirmed this by creating a new file on the remote machine and then verifying that the file existed on the remote machine.



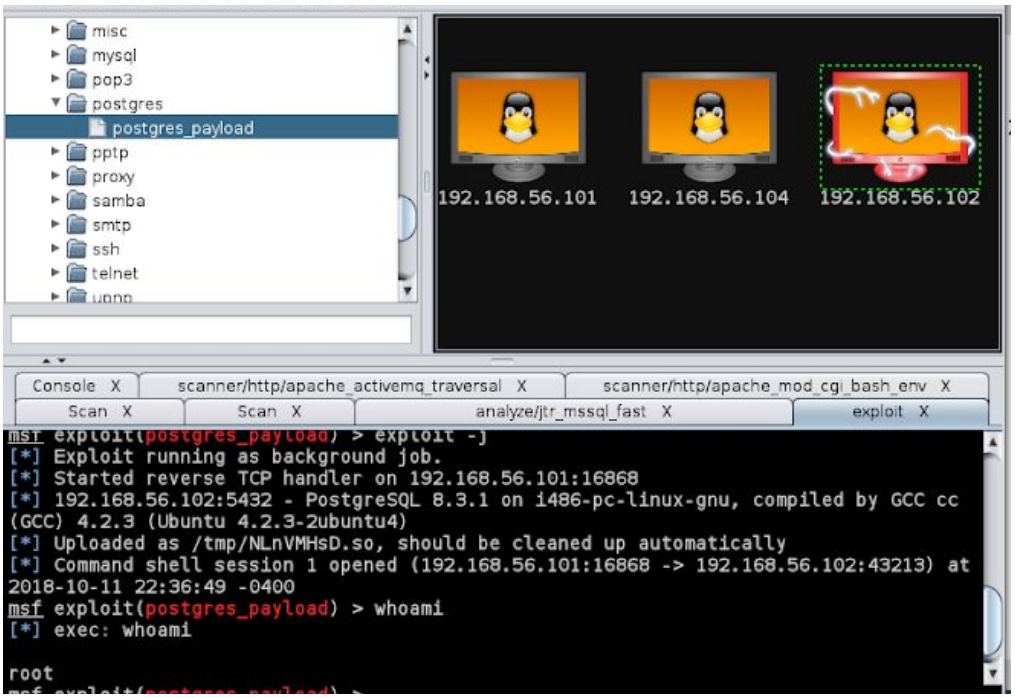
Listing and creating a file from the remote machine in the 'tmp' directory.



Listing the directory on the metasploitable machine before and after creating the new file on Kali.

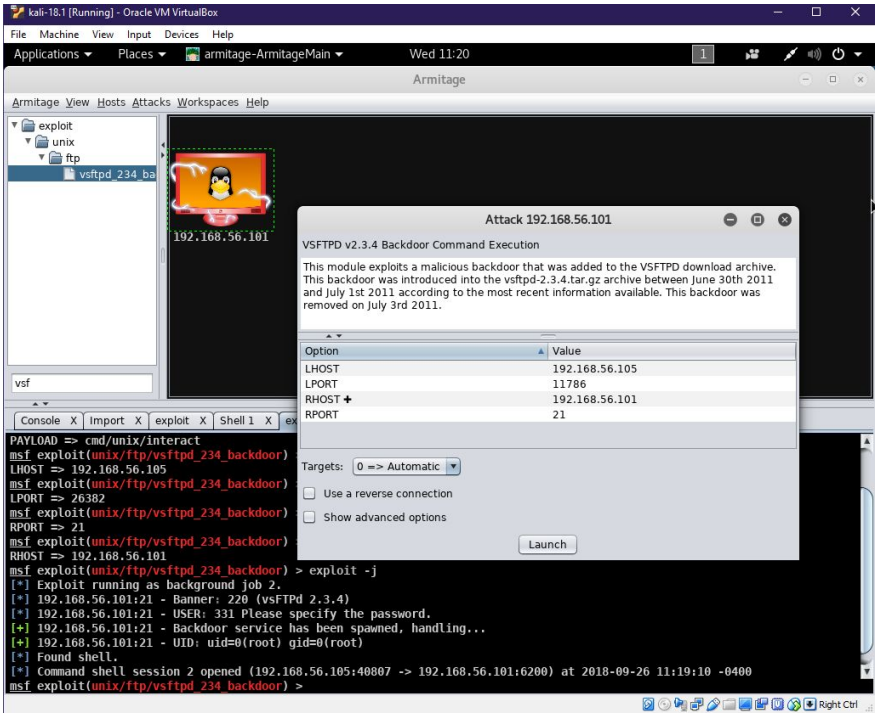
We checked the internet to learn that the Unreal IRC daemon version that this exploits (3.2.8.1) is from 2009, when a maliciously modified version of the IRC client was distributed on mirror sites. We view this as a reminder of the importance of software authentication.

Postgres Payload Exploit



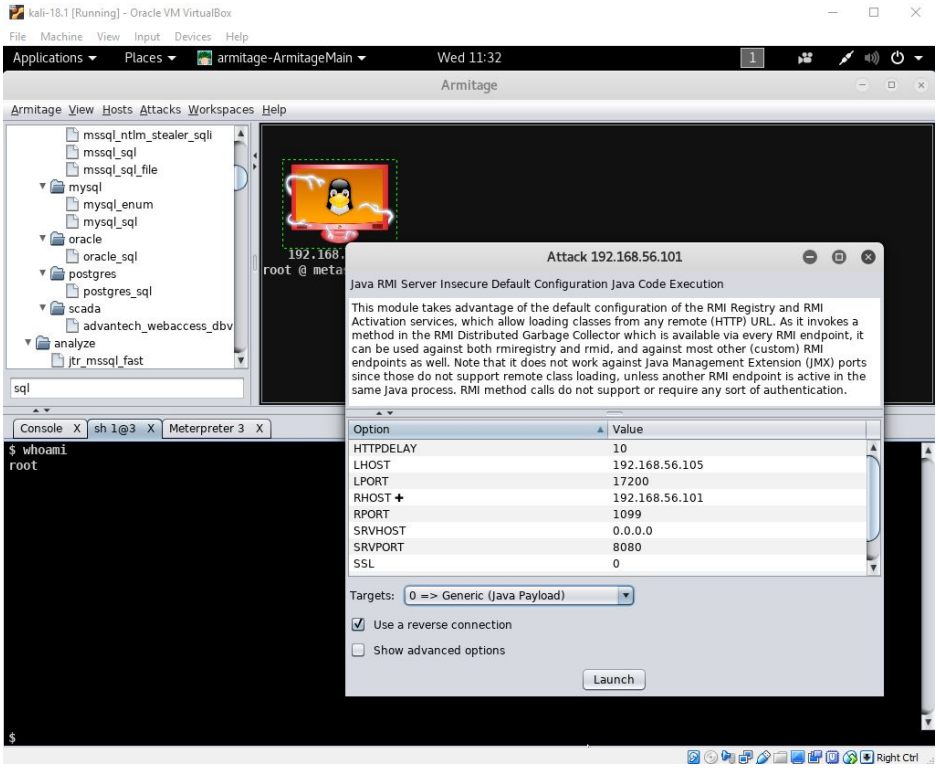
We obtained shell access using the Postgres Payload exploit. https://medium.com/@netscylla/postgres-hacking-part-2-code-execution-687d24ad2082 explains that if PostgreSQL has access to the system's `tmp` directory, as it does in some older default configurations, an attacker can then gain access to the file system via PostgreSQL. We were able to confirm that we had access to the entire file system and not just the `tmp` directory by changing to and listing different directories in the Meterpreter shell.

VSFTPD 2.3.4 Backdoor Command Execution



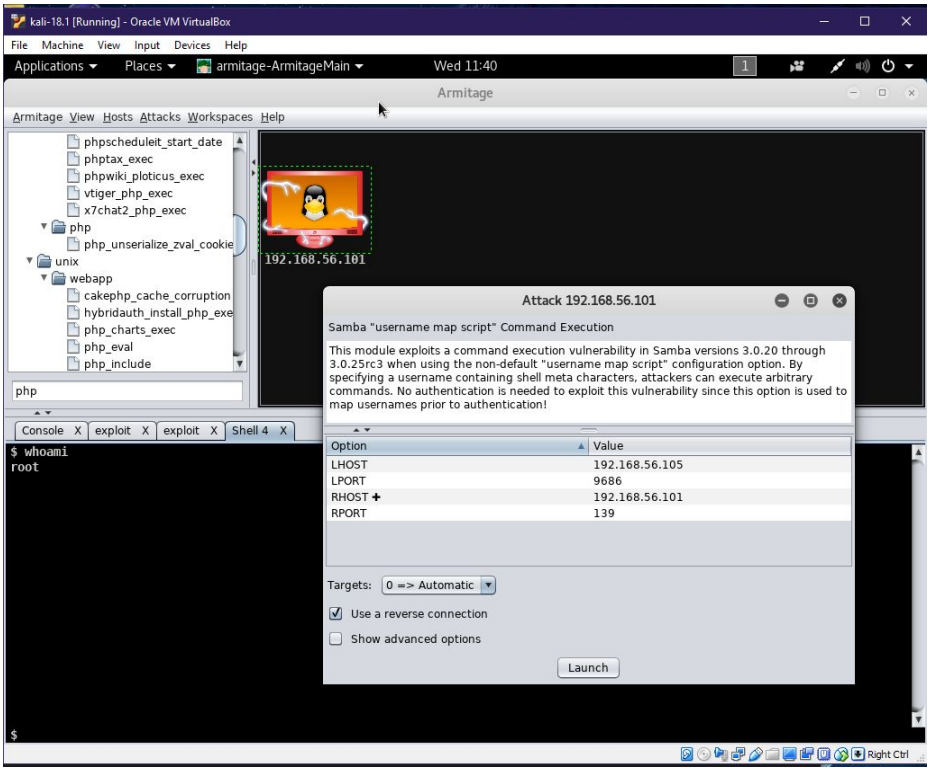
We obtained root access by launching an attack against the VSFTPD FTP server on the metasploitable machine. https://security.appspot.com/vsftpd.html Rapid7 notes that this backdoor was added maliciously to the version 2.3.4 client download in June 2011, and was existed until July 2011. https://www.rapid7.com/db/modules/exploit/unix/ftp/vsftpd_234_backdoor

Java RMI Server Insecure Default Configuration



We obtained a root shell by launching an attack that used Java remote method invocation. https://en.wikipedia.org/wiki/Java_remote_method_invocation. According to https://knowledgebase.progress.com/articles/Article/How-to-prevent-Java-RMI-class-loader-exploit-with-Admin Server, before version 7.2.1 of Java its RMI API's default configuration allowed for classes with malicious code to be uploaded and invoked remotely, but since that time (October 2012) the default configuration has been to disallow this. This is an example of why all software should always default to the secure option.

Samba “Username Map Script” Command Execution



A root shell was obtained by using this exploit that existed on an older version of the networking service Samba (https://whatis.techtarget.com/definition/Samba). 3.0.20 was released in 2005 with a vulnerability that allowed an attacker to execute commands using ‘shell metacharacters’ in their username, when a certain option was turned on. It appears that version 3.0.25, which corrected this issue, didn’t come out until 2007.

Overall Thoughts on Metasploit Exploitations:

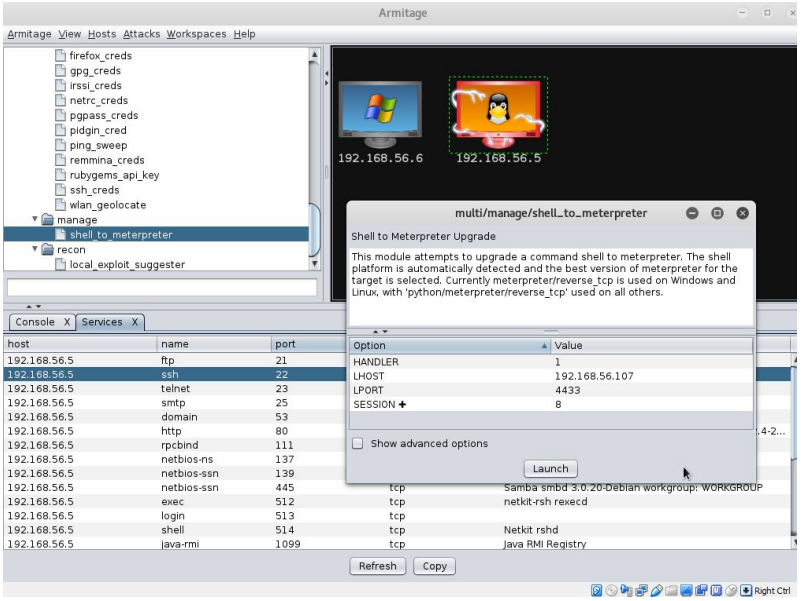
- Most vulnerabilities were short lived (many existed for less than a month).
- Most vulnerabilities were from a long time ago - the ones we used were from about 2007 to 2011.
- Many require the victim machine to have one or more configuration options set. In some cases the unsafe option is the default (as in the PostgreSQL and JAVA exploits). From a software design perspective, an obvious takeaway is that the default option should be the safe one. It also implies that care should be taken as to what software and what options are enabled on your system
- Some of the vulnerabilities were added by malicious actors providing altered versions of the software in question with the backdoors added. This would imply that the versions installed on the Metasploitable machine are not in fact genuinely from the publisher (at least insofar as they have specific outside modifications).

3.2.4 Harvesting Credentials from Metasploitable (10)

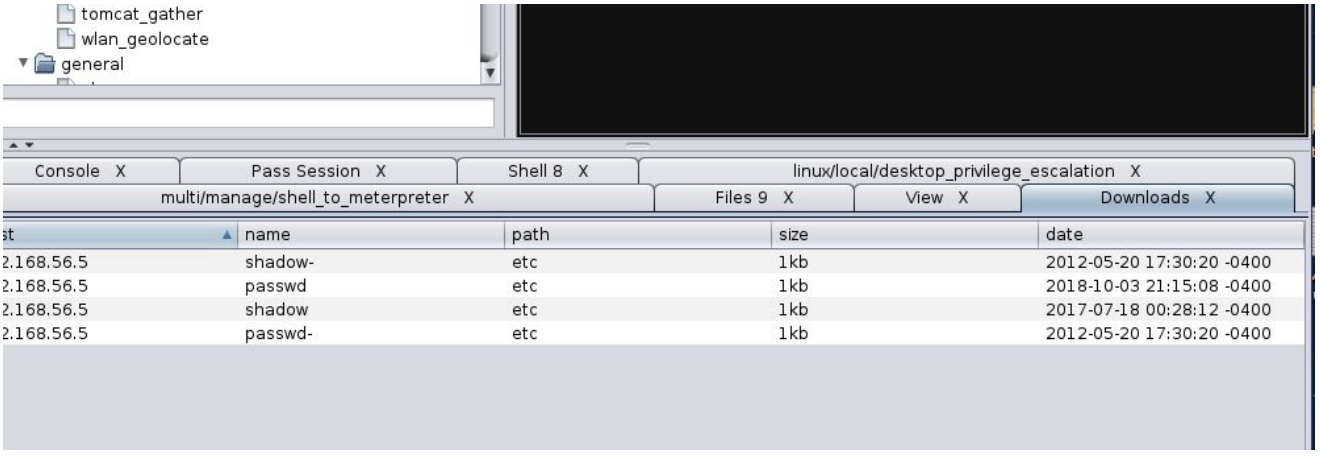
Our investigation into the credentials on the Metasploitable machine is as follows:

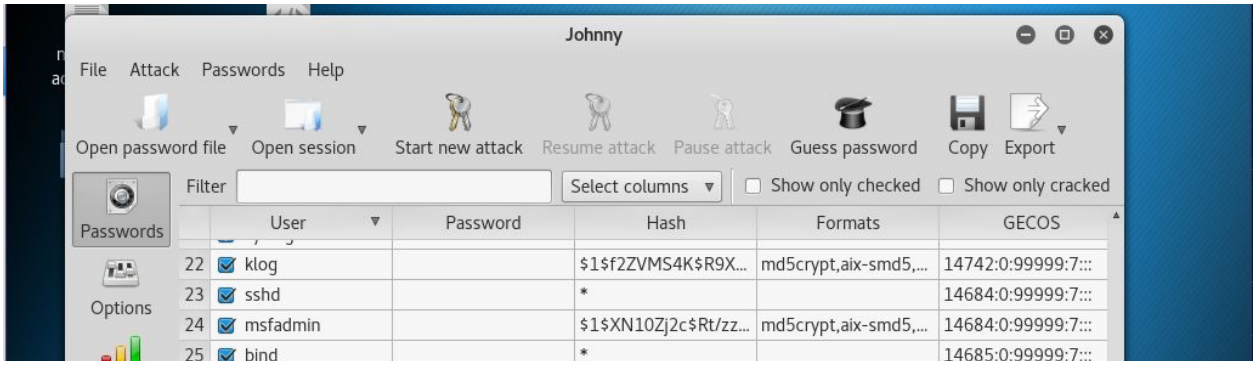
Obtaining and Cracking Shadow Password File

Running the exploit `vsftpd_234_backdoor` we gained a command shell as root, we then used the post exploit module `shell_to_meterpreter` to upgrade our command shell to a meterpreter to make life easier.

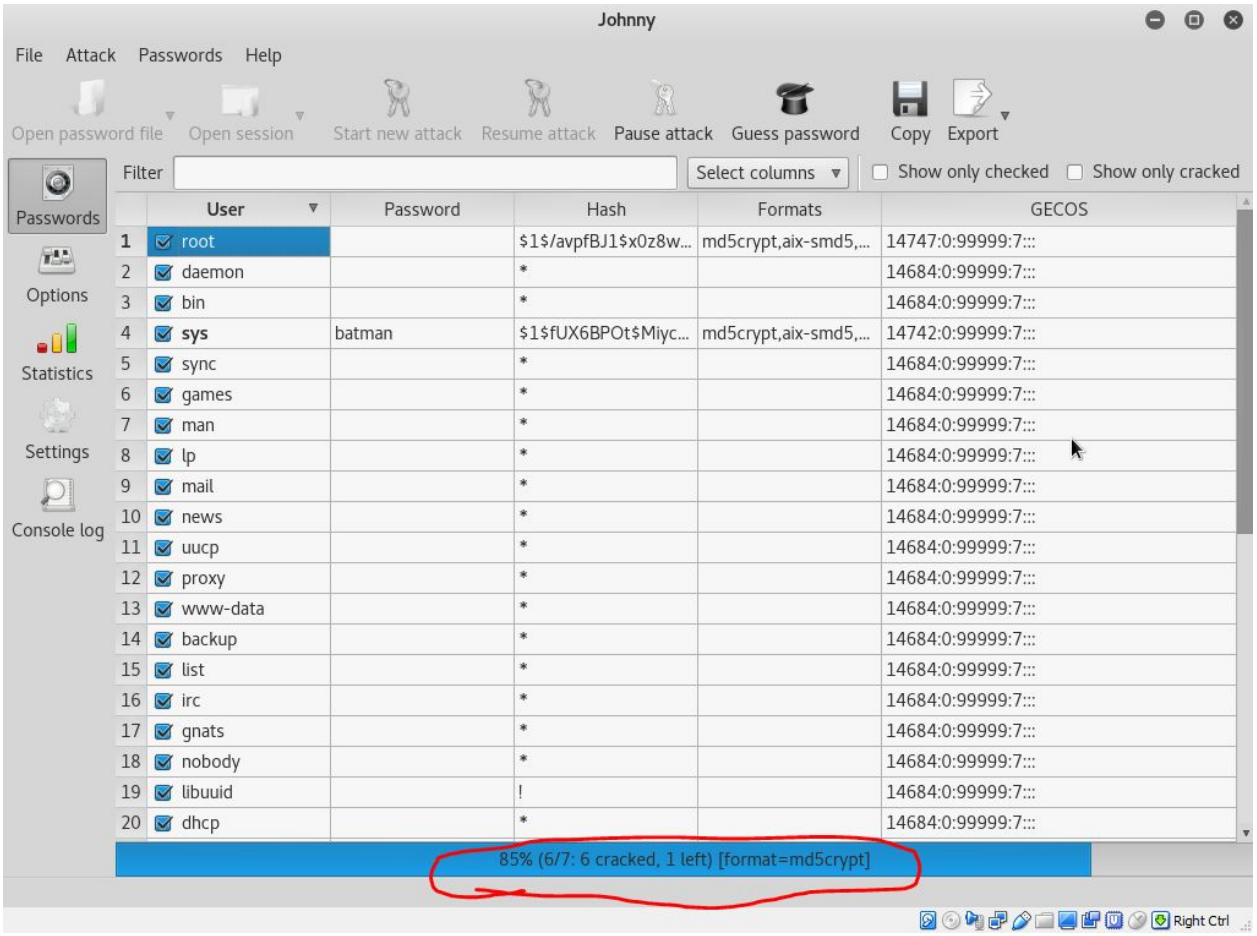


We downloaded the shadow password file using a meterpreter and ran it through jonny (the john the ripper gui)





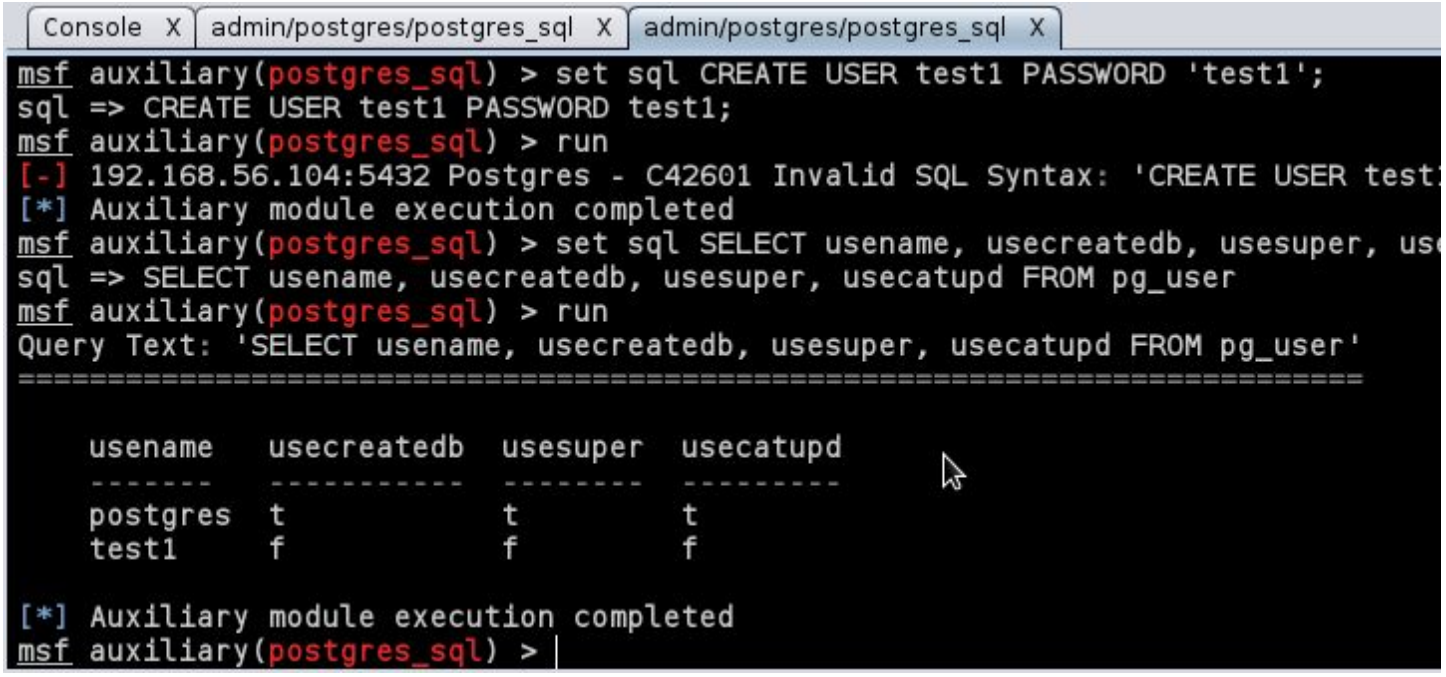
Within seconds we have cracked 6/7 of the md5crypt hashes.
Running this through hashcat on a GTX1080 for a few hours in brute force still did not crack the last hash.
We also tried a combination with the Rockyou wordlist without success.



Credentials we pulled:
(username, password)
(sys, batman)
(klog, 123456789)
(msfadmin, msfadmin)
(postgres, postgres)
(user, user)
(service, service)

Exploring the filesystem we were able to steal the SSH public and private keys. We believe we could use those to impersonate the computer in secure internet communications.
https://searchsecurity.techtarget.com/definition/Secure-Shell
https://www.pentestpartners.com/security-blog/how-to-abuse-ssh-keys/

Abusing PostgreSQL



Creating a new PostgreSQL user and listing user privileges on the Metasploit target in Armitage.

Knowing the PostgreSQL login enabled us to log in to the Postgre server on Metasploitable from Armitage.

We were then able to run SQL queries on the database. If there was sensitive information on this database, we would have access to it (there did not appear to be any information). We were able to list tables, create users, list version, etc. Experimenting with the **http://pentestmonkey.net/cheat-sheet/sql-injection/postgres-sql-injection-cheat-sheet** suggested to us that it should be possible to also gain access to local files, particularly on a version that is as insecure as the one on Metasploit, but Alister did not have any success doing this in the time we had.

If we had a specific goal in mind, we would be able to modify data on the tables. If we wished to cause damage, we could delete some or all tables or users.

Abusing FTP

We were able to log into the FTP server as anonymous.

Wondering what this would allow us to do, we found this article on the net
https://security.stackexchange.com/questions/184419/what-could-happen-if-someone-guessed-a-password-to-my-ftp-server
The obvious way we could abuse this is to steal or delete the files stored on the FTP server. A less obvious abuse is to upload unwanted material to the server, such as illegal files, which could be used to store the files or get the owner in trouble. It would also be possible to replace the owner's existing files with modified versions that contain malware or other harmful modifications.

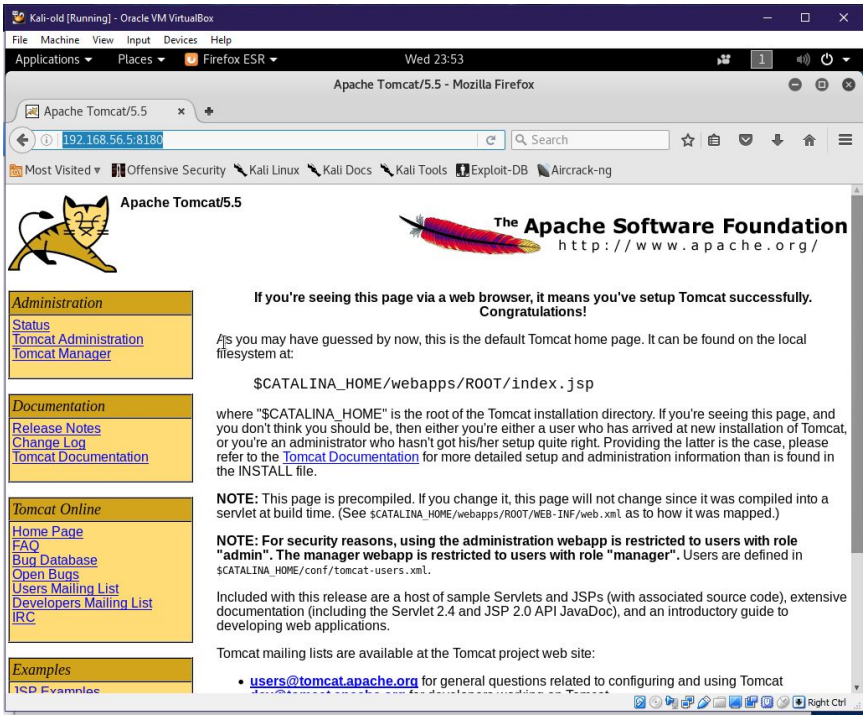
Abusing Apache Web Server

We noted that there was an apache server running on port 80.

host	name	port	proto	info
192.168.56.5	ftp	21	tcp	vsftpd 2.3.4
192.168.56.5	ssh	22	tcp	OpenSSH 4.7p1 Debian 8ubuntu1 protocol 2.0
192.168.56.5	telnet	23	tcp	Linux telnetd
192.168.56.5	smtp	25	tcp	Postfix smtpd
192.168.56.5	domain	53	udp	ISC BIND 9.4.2
192.168.56.5	http	80	tcp	Apache/2.2.8 (Ubuntu) DAV/2 (Powered by PHP/5....
192.168.56.5	rpcbind	111	udp	2 RPC #100000
192.168.56.5	netbios-ns	137	udp	Samba nmbd netbios-ns workgroup: WORKGROUP
192.168.56.5	netbios-ssn	139	tcp	Samba smbd 3.X - 4.X workgroup: WORKGROUP
192.168.56.5	netbios-ssn	445	tcp	Samba smbd 3.0.20-Debian workgroup: WORKGROUP

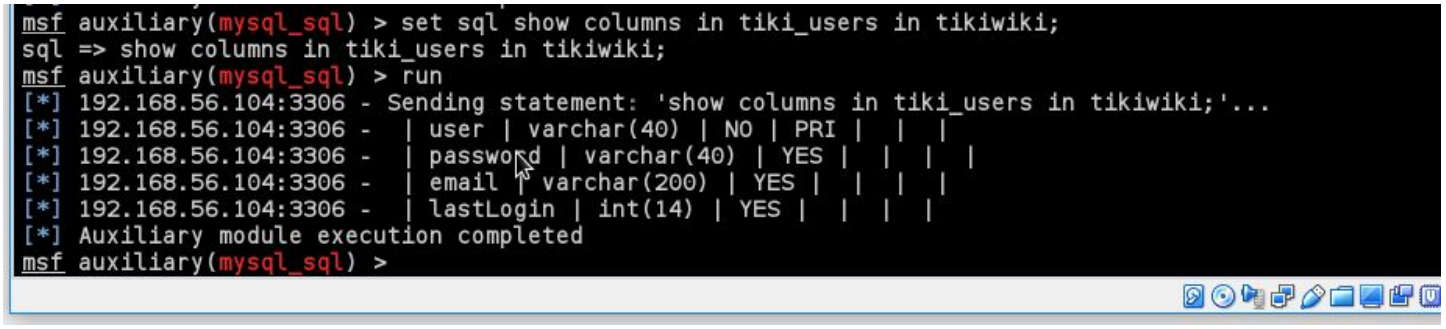
.Typing the address into the browser gets us into this server.

Which was also kind enough to list the Metasploitable admin login.
Another port on 8180 got us into this server:



The most obvious abuse here is that we would be able to modify the contents of any websites hosted on the server. In its most extreme form, our access could be used for illegal activities on the server.

Abusing MySQL



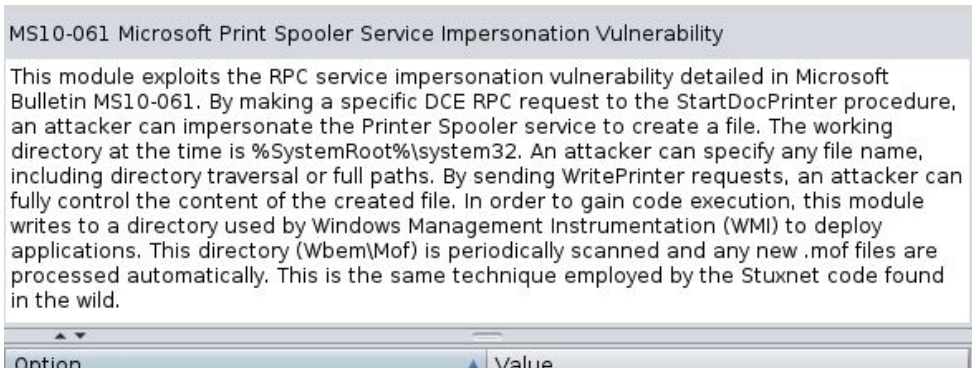
Listing columns of Tikiwiki tables on the Kali machine.

Finding the login and password for MySQL on Metasploitable enabled us to log into MySQL (using username: root and password: [blank]). We were able to run basic queries, finding out the list of databases, tables, and columns held in the MySQL server.
As an experiment, we were able to list the columns of the tiki_users table in the tikiwiki database, which would contain the password and emails of the users for this (presumably fictional?) company. Unfortunately we were unable to list table rows, though we assume that this is possible. The reason was that the Armitage module would execute a database query, then delete its instance at the end of that single query. MySQL seems to require that you change to a database before it lets you list the table columns, but because the instance is destroyed after one query your database change is reset as well. Presumably with enough research we could figure out how to solve this.

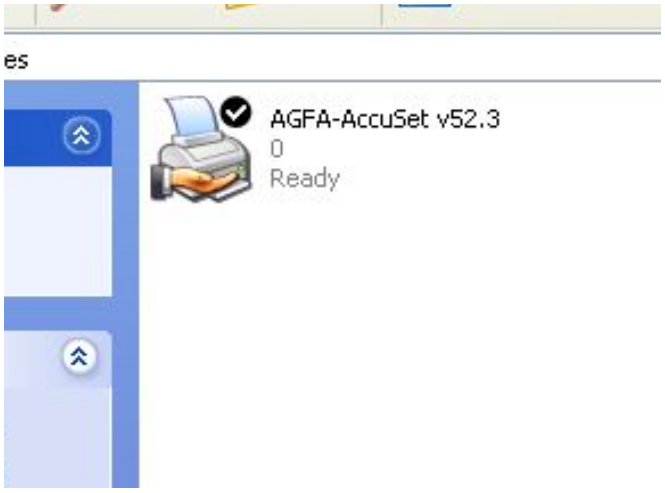
As with the PostgreSQL abuse, we could cause damage by deleting or altering databases or users.

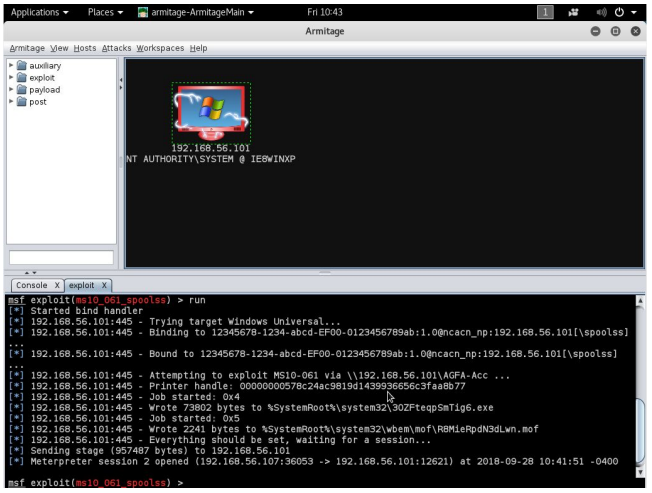
3.2.5 Exploiting Windows XP

Microsoft Print Spooler Service Impersonation

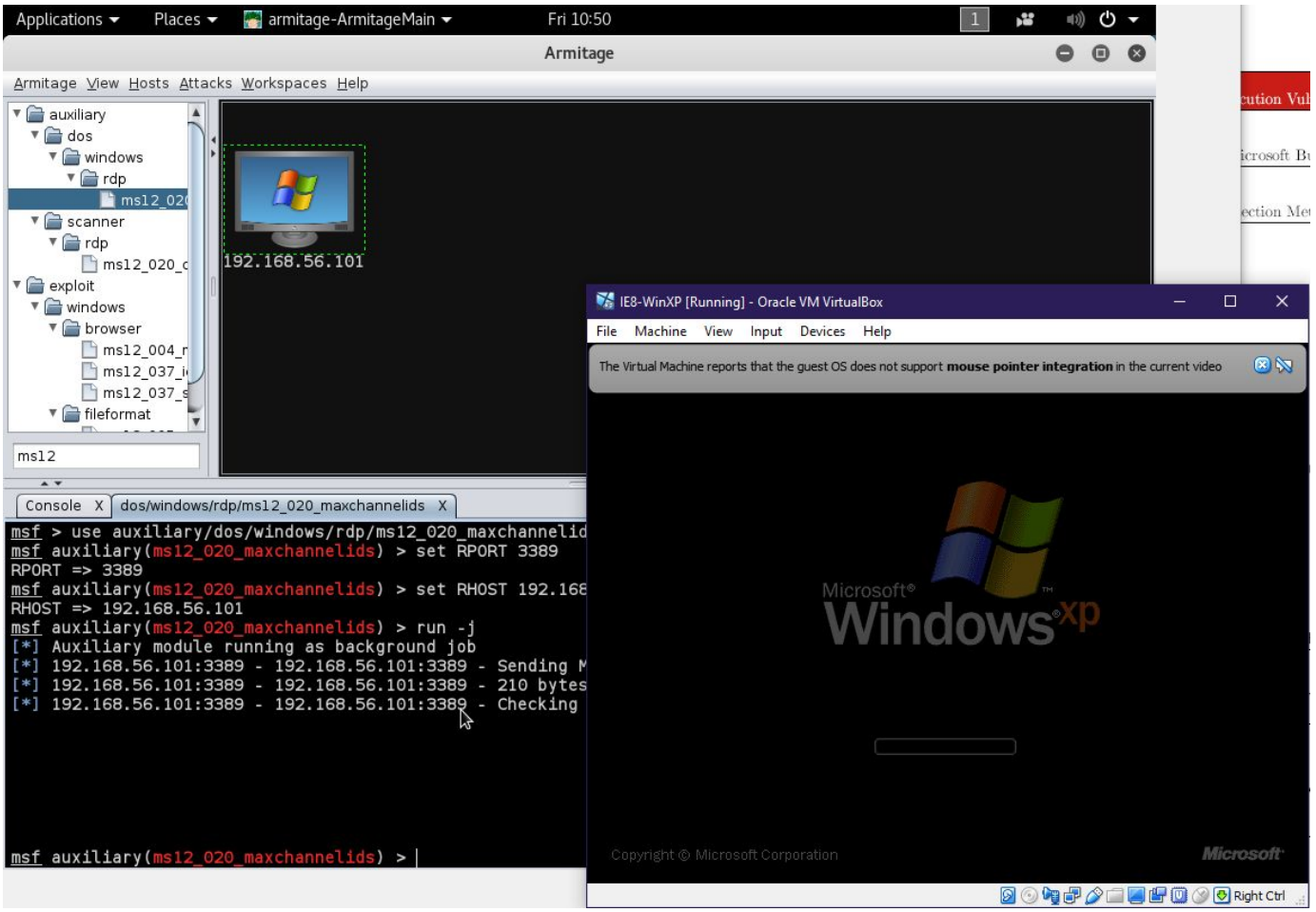


MS10-061 description states that we can attack the StartDocPrinter procedure this however does not work due to there being no printer service on the xp machine.
Since it is likely that a windows user would be using a local printer we have set up a local printer on the network, running the exploit now results in root access.





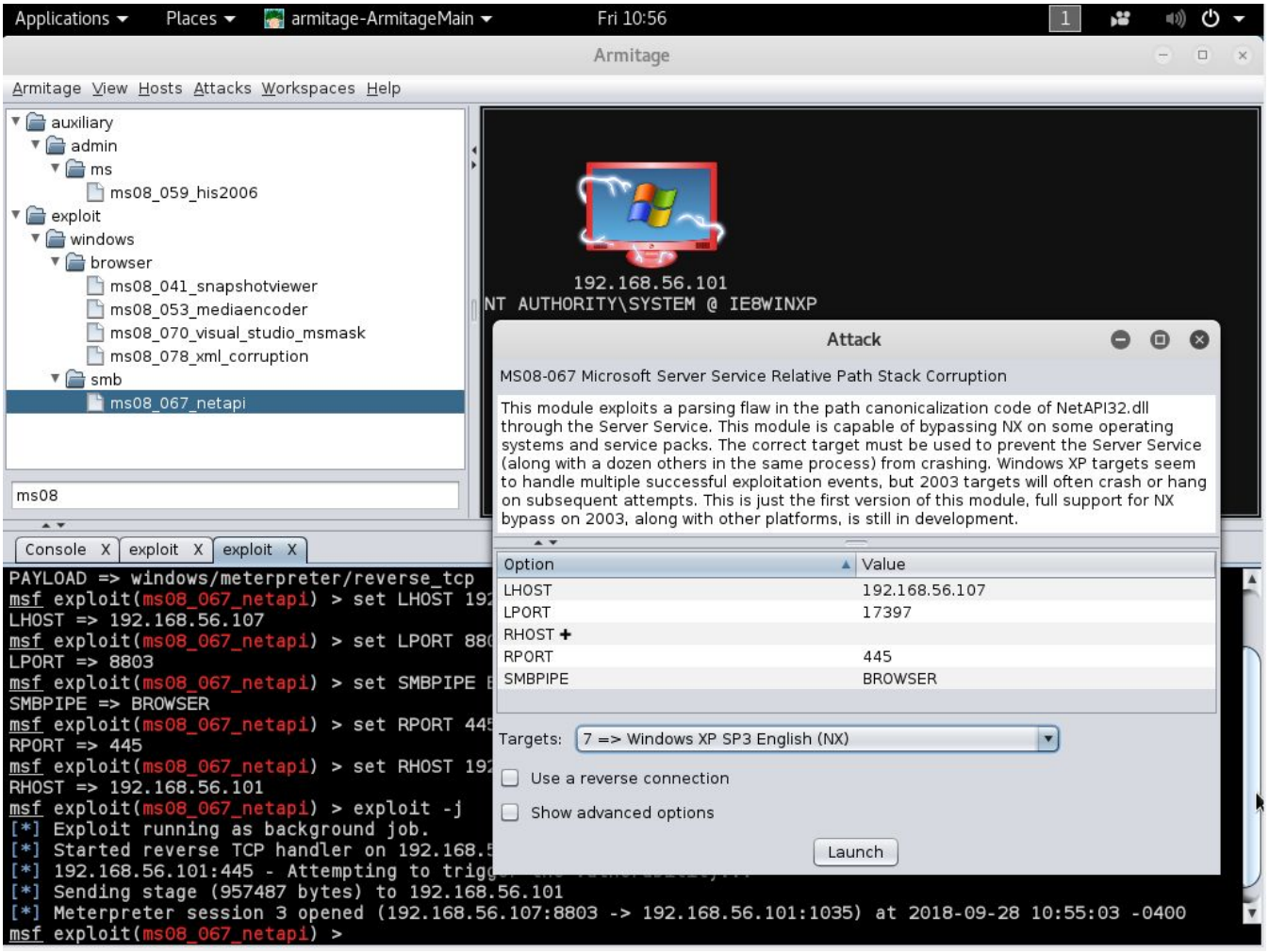
Microsoft Remote Desktop Protocol Remote Code Execution



We were able to crash the XP machine using this exploit.

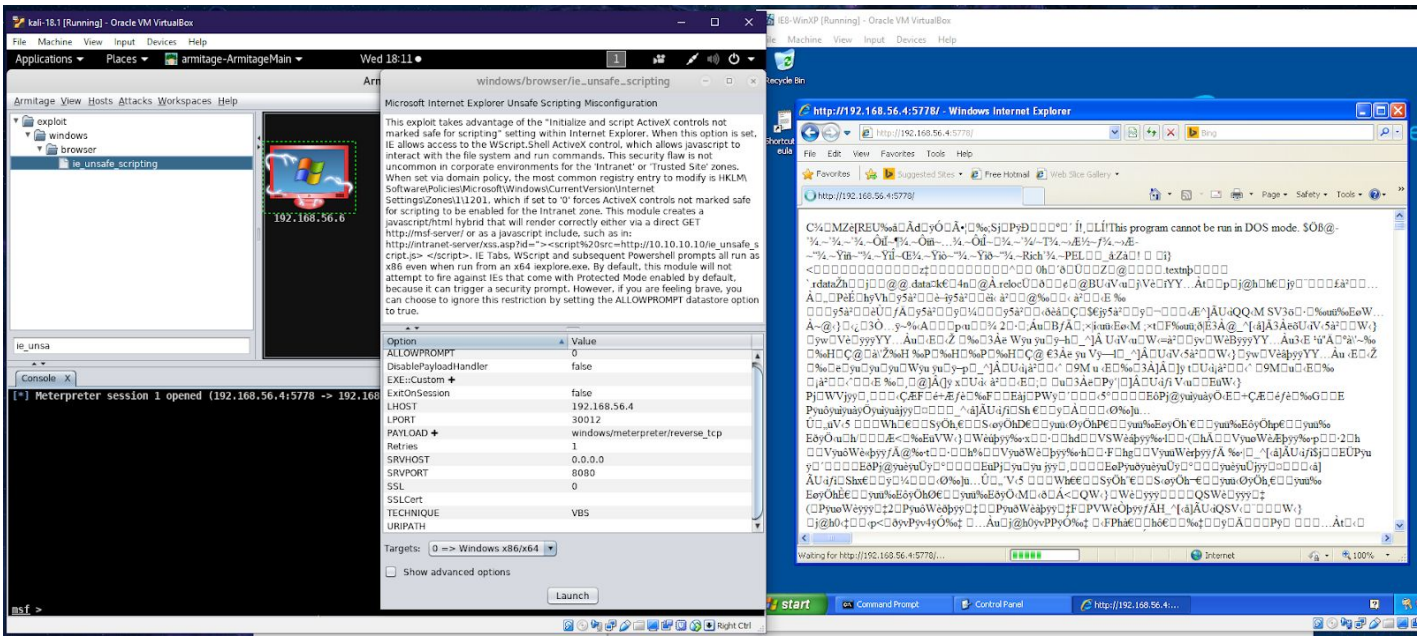
https://docs.microsoft.com/en-us/security-updates/securitybulletins/2012/ms12-020 from Microsoft explains the vulnerability as affecting the Remote Desktop Protocol on XP, and says that ‘specially crafted’ RDP packets sent to a machine could be used to execute malicious code. The OpenVAS report says the exploit can be used to run ‘arbitrary code or cause a denial of service’.
In the 2012 security bulletin regarding this exploit, Microsoft explained that Remote Desktop is not enabled by default on XP, which implies that disabling Remote Desktop is an obvious defense against this. A security update from 2012 also apparently changed the way ‘RDP packets were handled in memory’, preventing patched devices from being successfully exploited.
Further reading
https://security.stackexchange.com/questions/143485/is-there-a-rce-poc-for-the-cve-2012-0002-ms12-020 implies that the vulnerability was related to certain parts of the memory becoming exposed in the moment after a user disconnects, into which malicious code could be inserted.

Also related was this 2008 exploit, which gained us full root control:



https://docs.microsoft.com/en-us/security-updates/securitybulletins/2008/ms08-067 (It is listed as 'Vulnerability in Server Service could allow remote code execution' in OpenVAS), and targeted an older vulnerability in how the server service handled requests for the Remote Desktop Protocol. According to Microsoft, this vulnerability would allow an attacker to gain full control of the system using 'specially crafted RDP requests' and without needing any authentication.

Exploiting Internet Explorer



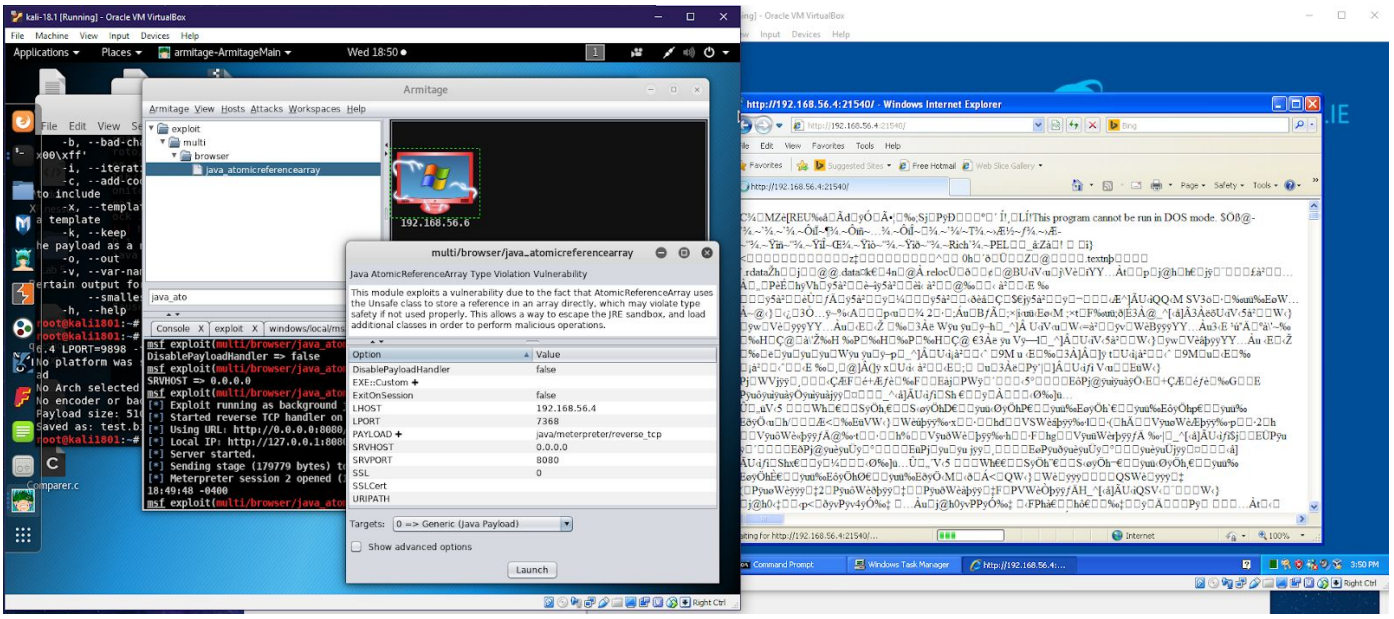
We first tried the well known AURA exploit against IE but this didn't work on IE8, however we discovered that "ie_unsafe_scripting" does work.
We assume we can social engineer the victim into visiting 192.168.56.4:XXXX, for example we could achieve this by sending the link in an email to the user.

EternalBlue

https://en.wikipedia.org/wiki/EternalBlue
This took a little bit to get working.
This required us to set up admin group and change local security policy to classic to prevent giving guest accounts.
There was also an issue with armitage not having an exploit for windows xp which we managed to workaround by setting the exploits verify target and `verifyArch` flags to false.
When we finally got the exploit to work we could not gain a shell however the XP display glitched out and then blue-screened.

Java Atomic Reference Array

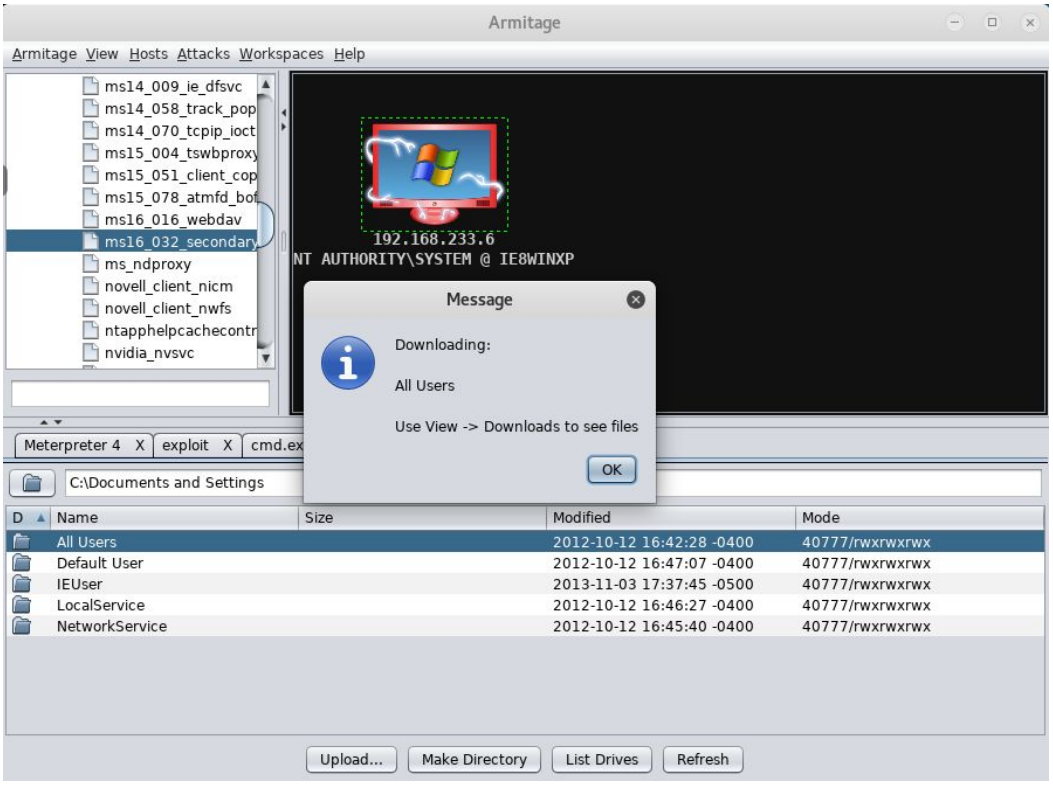
This exploit resulted in a root shell and also requires us to assume that we are able to trick the user into visiting a link.



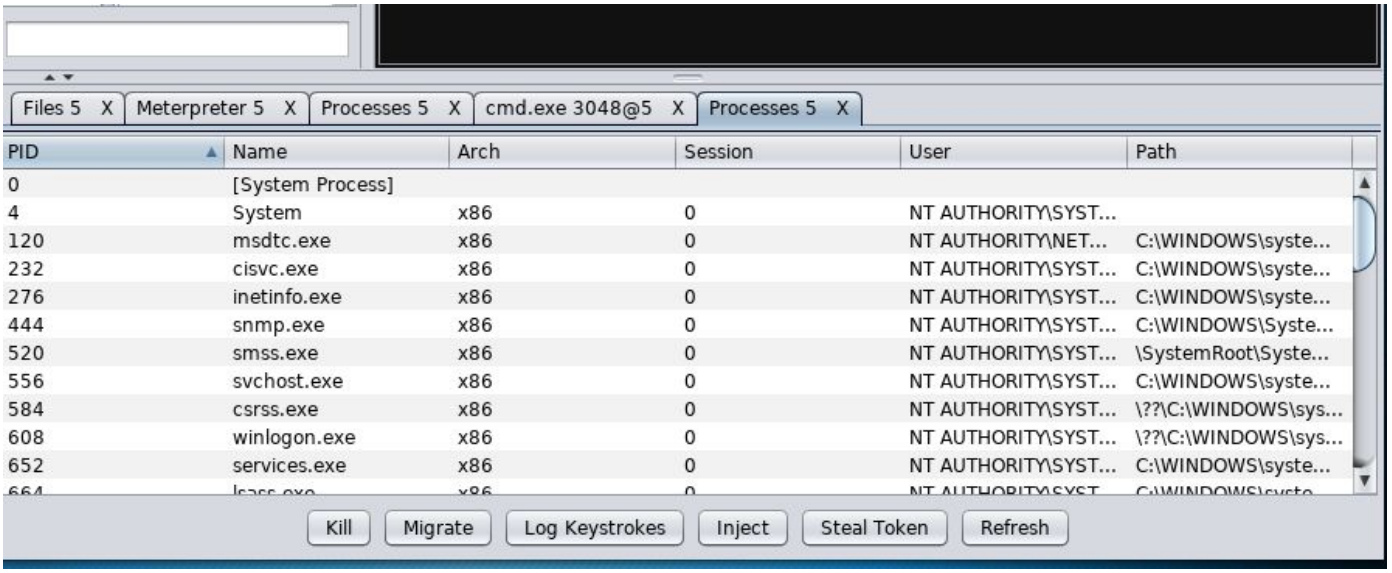
3.2.6 Exploring Post Exploitation Capabilities in Windows XP (10)

File System Access

Once we had root control we were able to upload and download any file we wanted. We could browse the directory structure. We were able to move, create or delete any files, even system files which would corrupt the OS. We could theoretically upload modified versions of the system files to obtain persistence.

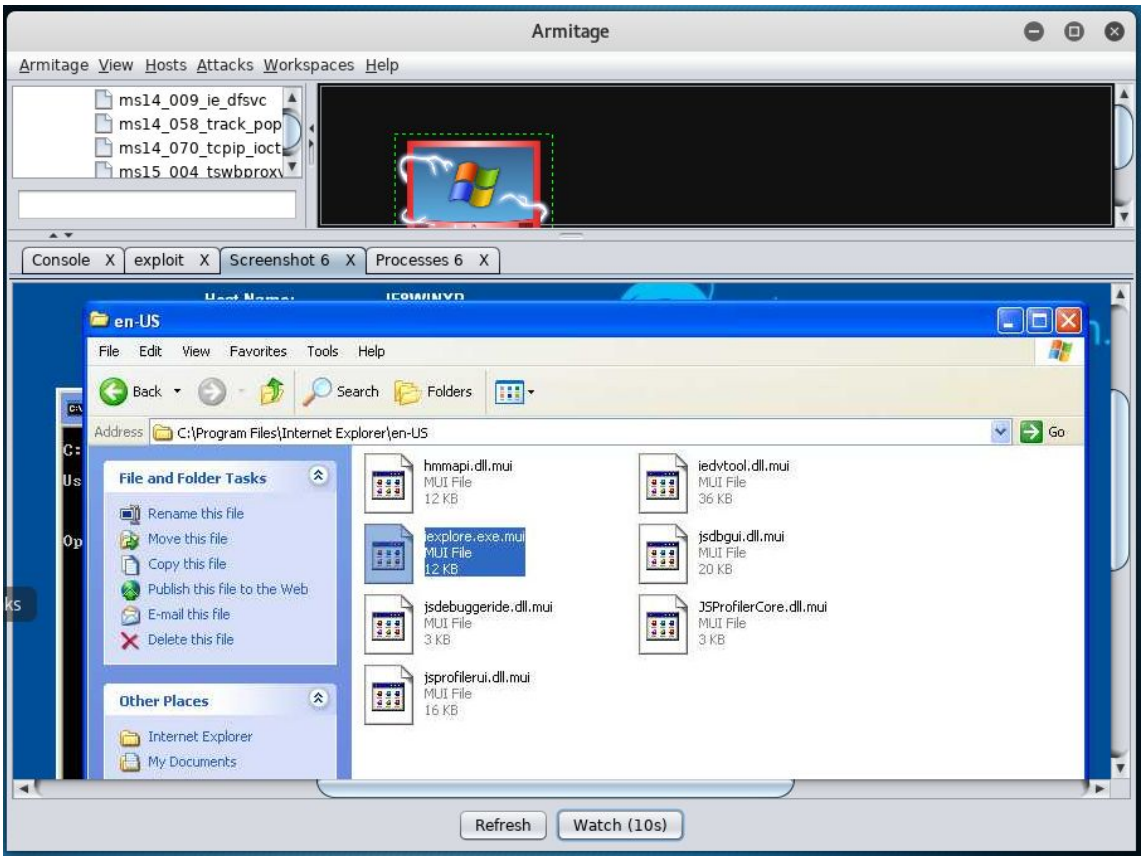


Process Modification



We were able to experiment with killing processes. Killing explorer removed the desktop interface in the XP machine.

Viewing the Screen

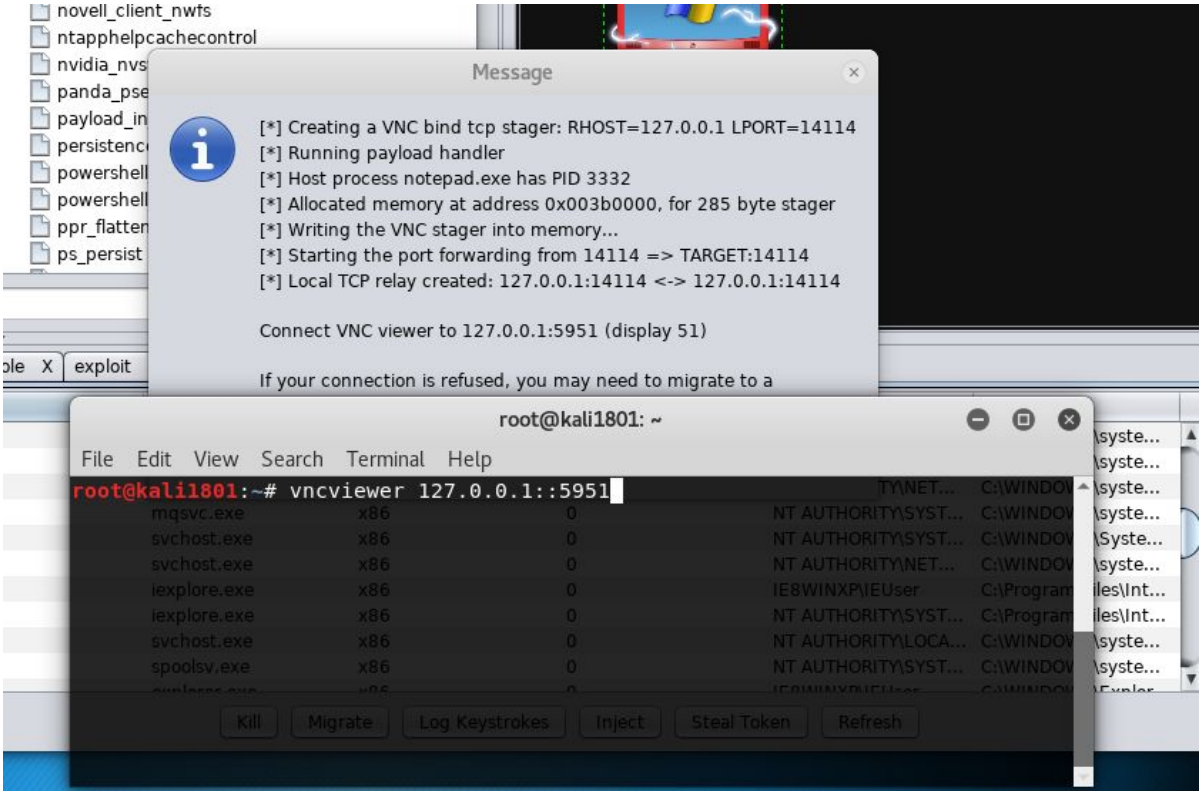


We used the screenshot option to allow us to see exactly what is on the screen on the XP VM. It updates every few seconds. This is obviously extremely dangerous, as it could be used to reveal personal information, including information that would let somebody discover things like banking details, logins, and even clues to your physical location. If you had encrypted documents, the encryption would be useless if an attacker could view your screen while you opened the document.

Webcam Access

We note that we would be able to access the remote machine's webcam. This is clearly dangerous in terms of privacy and confidentiality. It could also reveal your physical location to an attacker. The VM did not have a webcam so we were unable to test this.

VNC connection

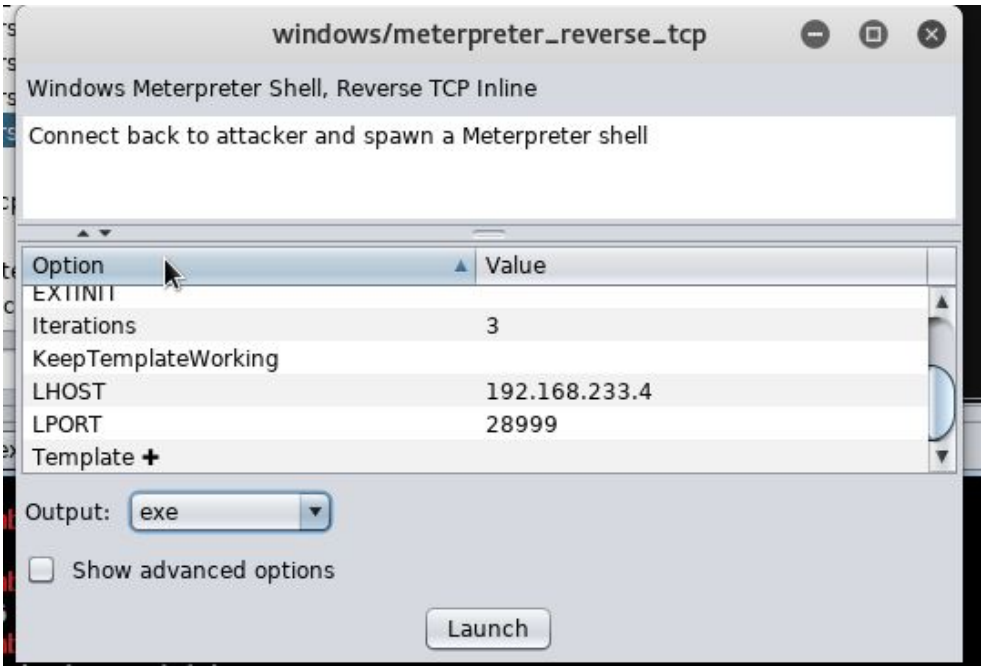


Using a VNC connection allowed us to control the remote machine in real time. We could see the changes happening on the remote machine. Unfortunately after a few seconds this was also crashing the remote machine. If we were so inclined, we could have tried to fix this problem.

Recording Keystrokes

This is probably among the most dangerous abuses as it allows an attacker to work out your logins, passwords, and secret information without having to decrypt them. It could give them access to your bank accounts, physical address, emails and private documents.

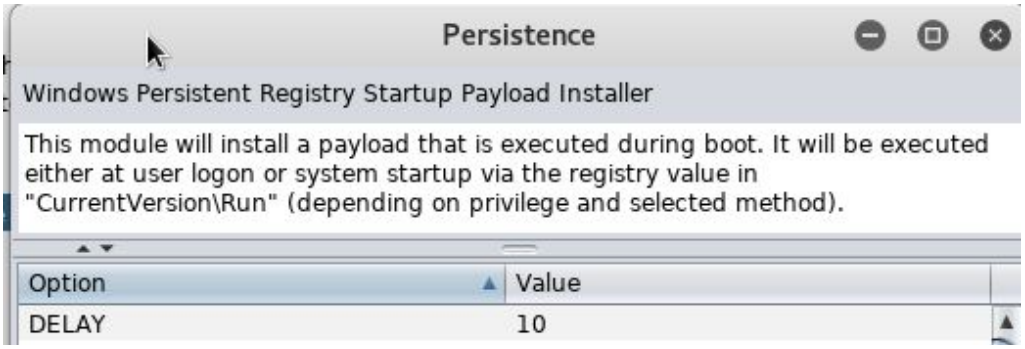
Gaining Persistence



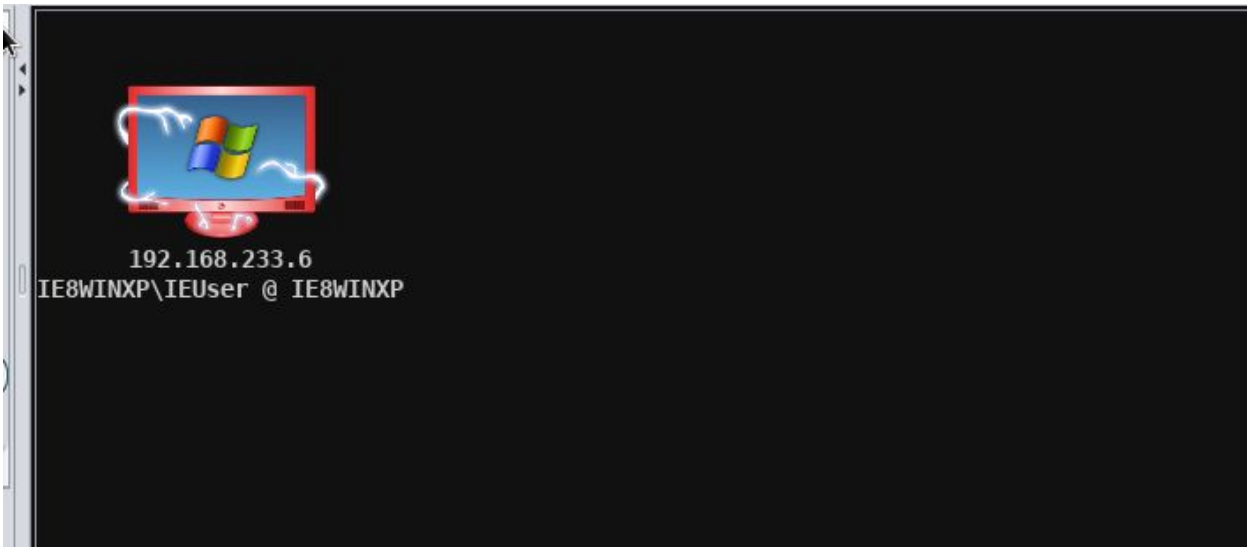
We explored gaining persistence by creating a program to use a reverse_TCP meterpreter exploit to contact kali and open a meterpreter shell when the XP VM starts up. Lewis called it `happydays.exe`.

To deploy it, we were able to upload it to the file system as we already had root access. But we could have tricked a user into clicking a script, opening it as an email attachment, we could also disguise the file, or put it on a USB drive and leave it somewhere obvious, and so on.

When we set up a reverse listener on port 28999 in Armitage, we got a meterpreter shell when the happydays.exe was run from the XP machine by the user. A simple way to get it to run on its own would be to put it in the startup directory. Some research on the net also informed us that adding a registry entry to `HKLM\Software\Microsoft\Windows\Current Version\Run` would also allow us to run the program on startup.



We discovered that this is what Armitage can do for us using the persistence module, allowing us to gain persistence very easily on the victim machine.

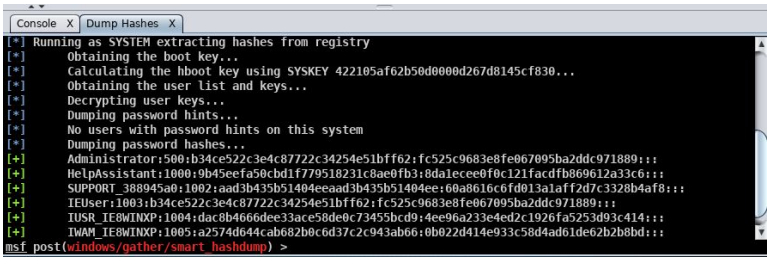


After running the Windows Persistent Registry Startup Payload Installer, we get a shell every time the VM boots.

Some Other Random Things

We can execute arbitrary commands!
It is possible to record the xp microphone, however it does not have one.
We managed to change the desktop wallpaper 😊

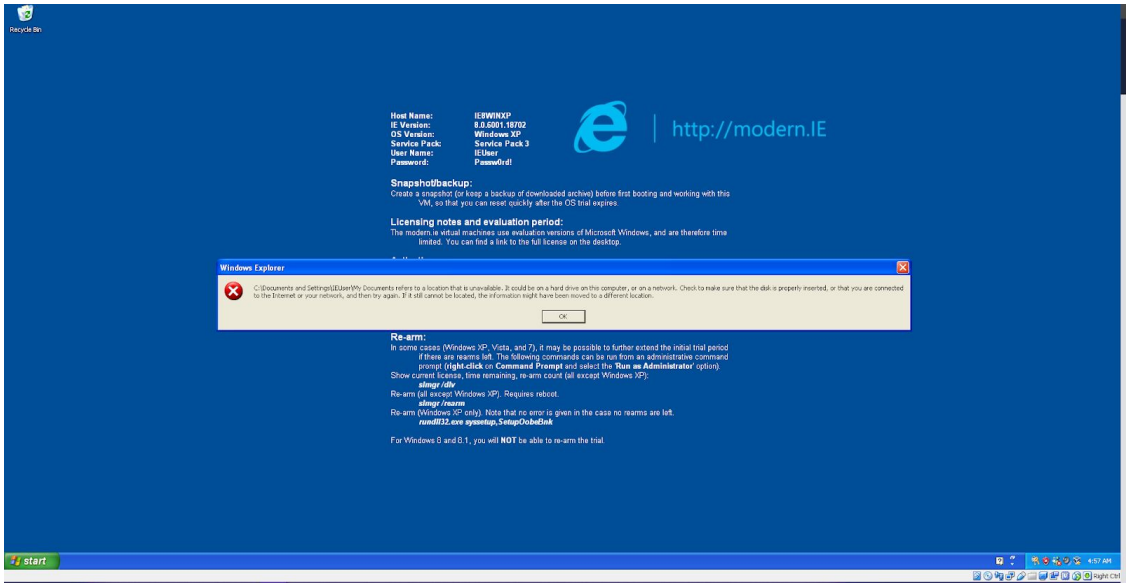
We can steal the password hashes using the dump hashes module and then run them through john the ripper or hashcat:



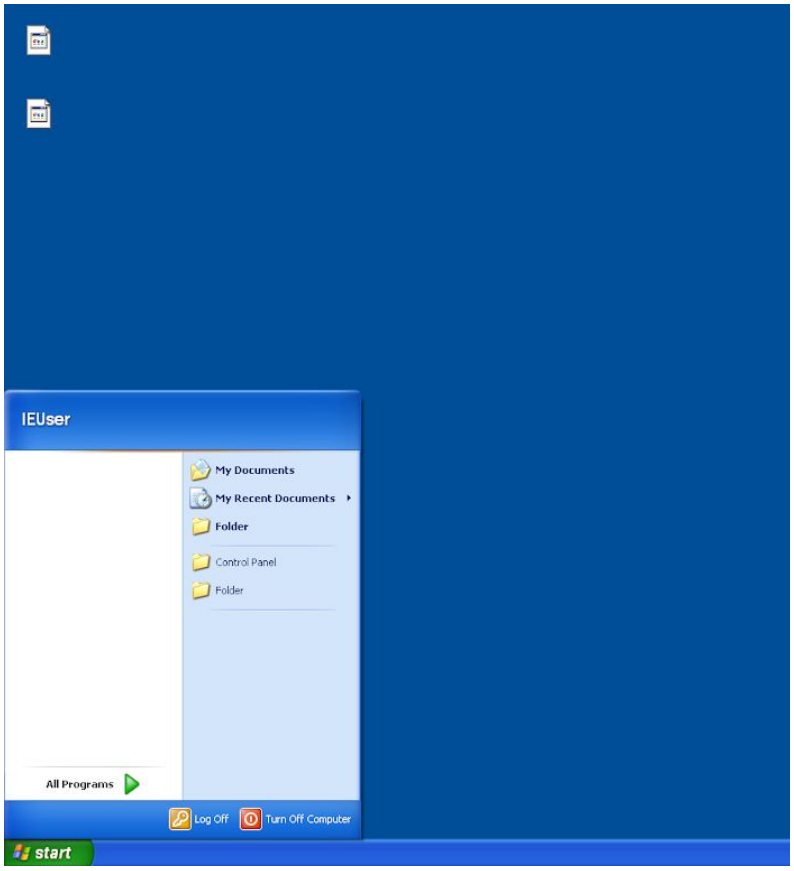
Completely Wrecking the VM

We decided to send the xp vm out using the command: `rd C: /s /q`.

5 mins in we have no files!



About 10 mins in



Can’t boot the vm anymore the MBR is gone.☹

3.3 Detection and Prevention (40)

3.3.1 Perform a Port Scan with Nmap

Nmap scans used
-p- Scan all ports (not used on udp scans due to time)
-sS SYN scan
-sT TCP connect scan
-sU UDP scan
-sN Null scan
-sX Xmas scan
--scanflags FIN
--scanflags SYN FIN

nmap -sU (UDP scan)

nmap -sS -p- (TCP SYN scan with full port range)

nmap -sT -p- (TCP connect scan with full port range)

```
Not shown: 987 closed ports
PORT     STATE SERVICE
25/tcp   open  smtp
80/tcp   open  http
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
1028/tcp open  unknown
1801/tcp open  msmq
2103/tcp open  zephyr-clt
2105/tcp open  eklogin
2107/tcp open  msmq-mgmt
3389/tcp open  ms-wbt-server
```

nmap -sX (xmas scan - FIN URG PSH flags set)

```
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS
system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.171.4
Host is up (0.00047s latency).
All 1000 scanned ports on 192.168.171.4 are closed
MAC Address: 08:00:27:05:42:F3 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1.40 seconds
```

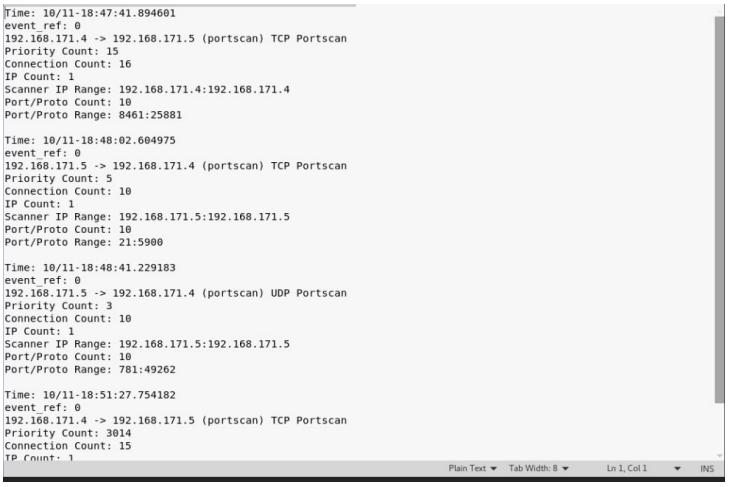
3.3.2 Mitigate Port Scanning With Snort (10)

Configuring Snort:
Installed and altered config file, set home net to 192.xxx.xx.0/24, also set rule directories and log directories.

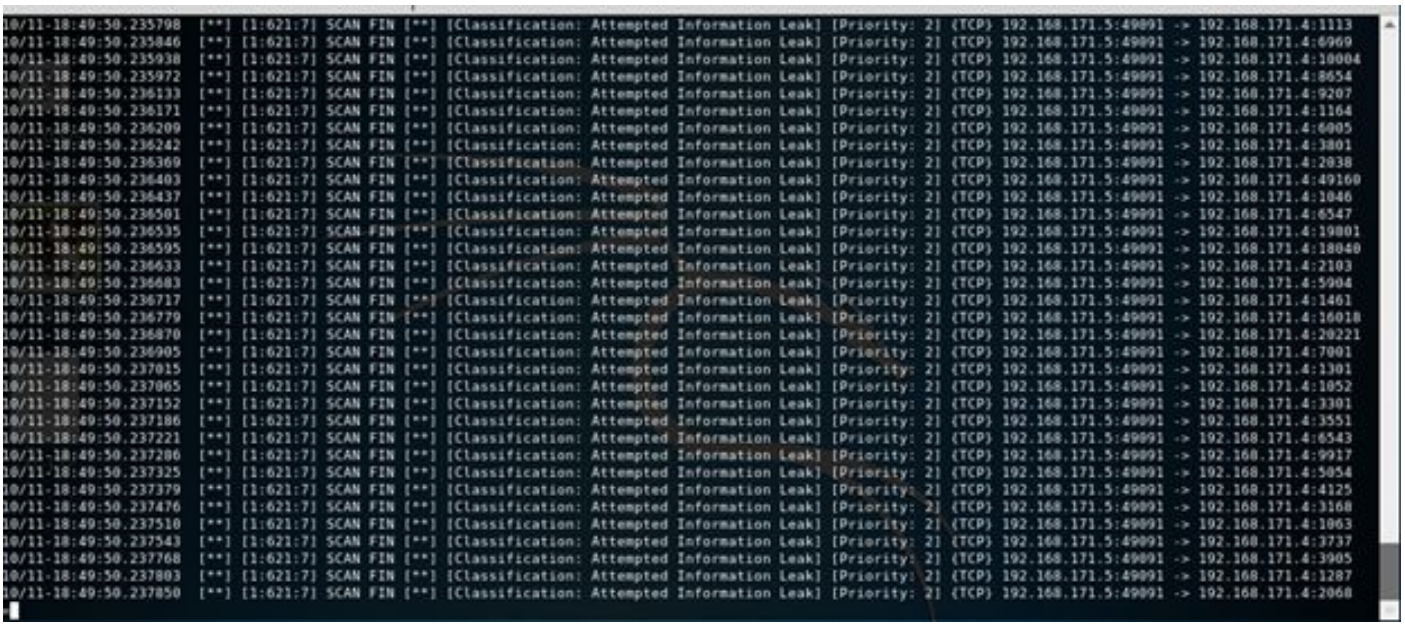
We enabled the preprocessor, increased the memcap and sense level so that we would detect most tcp scans, we also set the logfile for the alerts.

```
# Portscan detection.  For more information, see README.sfportscan
preprocessor sfportscan: proto  { tcp } memcap { 20000000 } \
                         sense_level { high } scan_type { portscan } \
                         logfile { tcpScanlog.log } detect_ack_scans
```

The preprocessor is now working (see the log file below) however, it is not detecting any of the following scan types: FIN, NULL, SYN FIN, FIN PSH URG



Because some scans are still not being detected we further configured snort by Including the default “scan.rules” set this allowed snort to detect any portscans over the network but this was ineffective because any single packet that matches the rules will be logged, this resulted in overloaded log files and console.



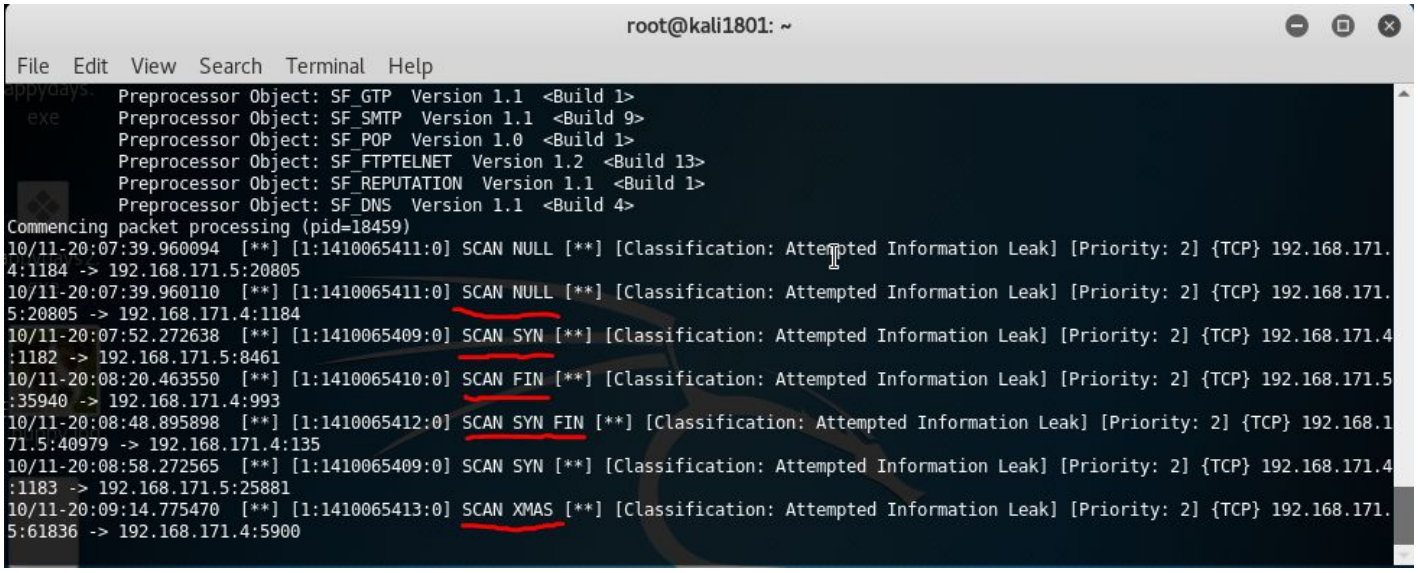
Swamped with alerts.

We wrote some rules to detect these scans in a more efficient way:

```
# Detect nmap tcp scans
alert tcp any any -> any any (msg:"SCAN SYN"; flags:S; classtype:attempted-recon; sid:10000000001; threshold: type both, track by_src, count 20, seconds 60;)


# Detect nmap fin scans
alert tcp any any -> any any (msg:"SCAN FIN"; flags:F; classtype:attempted-recon; sid:10000000002; threshold: type both, track by_src, count 10, seconds 30;)

# Detect nmap null scans
alert tcp any any -> any any (msg:"SCAN NULL"; classtype:attempted-recon; sid:10000000003; threshold: type both, track by_src, count 1, seconds 60;)

# Detect nmap syn fin scans
alert tcp any any -> any any (msg:"SCAN SYN FIN"; flags:SF; classtype:attempted-recon; sid:10000000004; threshold: type both, track by_src, count 10, seconds 60;)


# Detect nmap xmas scans
alert tcp any any -> any any (msg:"SCAN XMAS"; flags:FPU; classtype:attempted-recon; sid:10000000005; threshold: type both, track by_src, count 5, seconds 60;)
```

These rules are similar to the ones in scans.rules however we have set thresholding to both limit the number of alerts and avoid as many false positives by requiring a certain number of signature matches from a given src in a set time.

The result is as follows after allowing nmap to run each scan as listed above, as you can see we can now detect all the scans and avoid filling up our logs (although these are still rough and wouldn't be ideal for real implementation).

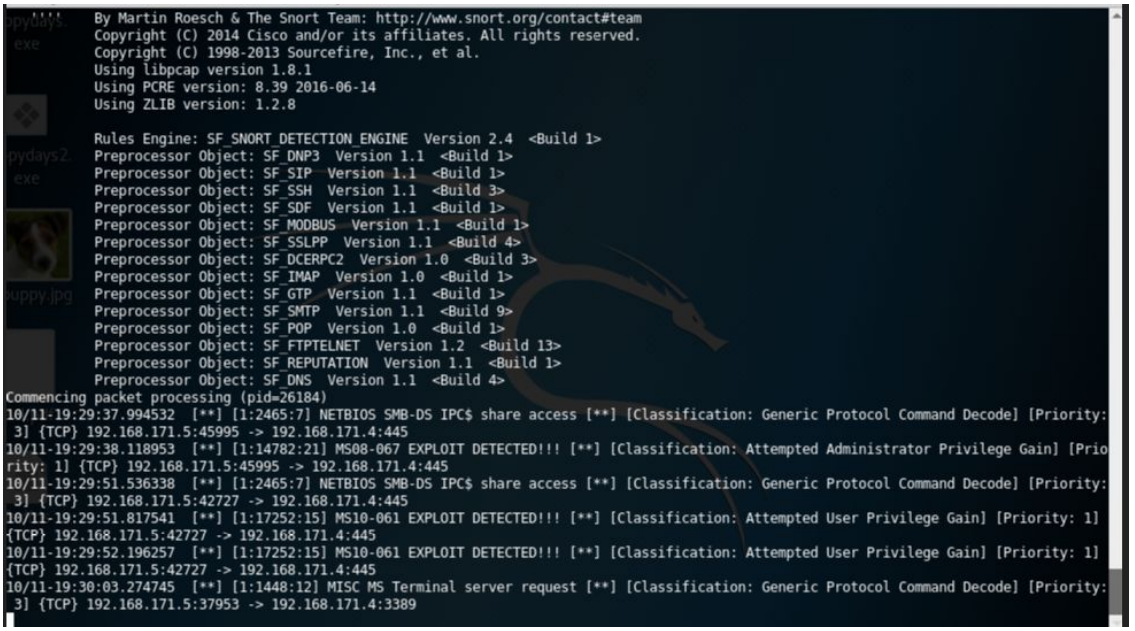


3.3.3 Mitigate Exploitation with Snort (5)

We attempted to write our own rules by analyzing packets with wireshark while sending exploits, this was not successful.
We searched for rules that related to the exploits we used on Windows XP. We found a large repository of rules specifically targeted at windows xp. Searching those rules for keywords relating to the exploits we had used turned up a fair number of hits. We tried a few and had to relax some of the attributes as they were too specific and managed to detect some of the exploits.

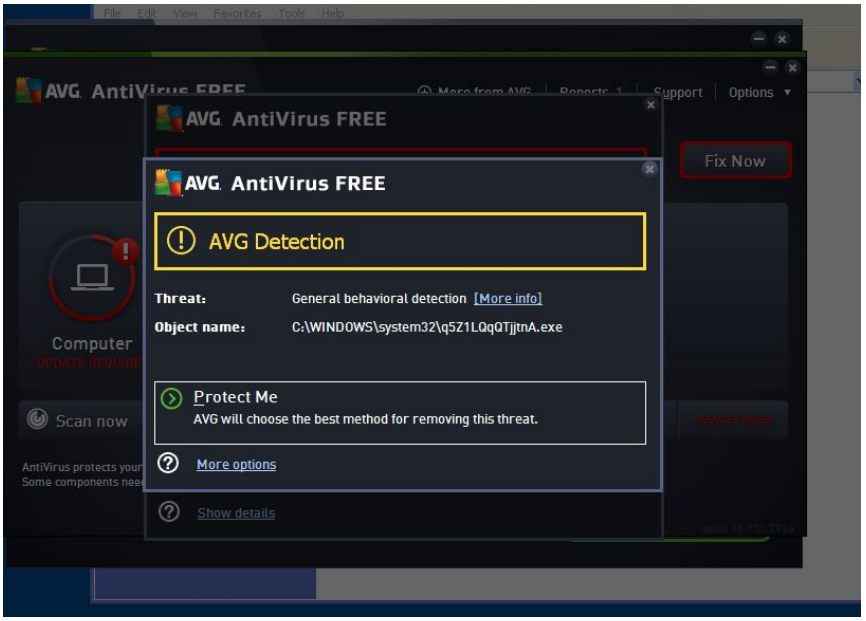
```
# Detecting ms10-061
alert tcp any any -> any any (msg:"MS10-061 EXPLOIT DETECTED!!!"; flow:to_server,established; dce_iface:12345678-1234-abcd-ef00-0123456789ab; dce_opnum:17;
 dce_stub_data; content:"|01 00 00 00|"; depth:4; offset:20; pcre:"/\x2E\x00(e\x00x\x00e|d\x001\x001\x00|m\x00o\x00f\x00|s\x00h\x00n\x00)/iR";
metadata:policy max-detect-ips drop, policy security-ips drop, service netbios-ssn; reference:cve,2010-2729; reference:url,technet.microsoft.com/en-us/security/bulletin/MS10-061;
 classtype:attempted-user; sid:17252; rev:15;)


# Detecting ms12-020
alert tcp any any -> any any (msg:"MS12-020 EXPLOIT DETECTED!!!"; flow:to_server,established; content:"|02 F0 80 7F 65|"; content:"|03 00|"; within:2; distance:-9;
detection_filter:track by_src,count 10,seconds 3; metadata:policy balanced-ips drop, policy max-detect-ips drop, policy security-ips drop; reference:cve,2012-0002;
reference:url,technet.microsoft.com/en-us/security/bulletin/ms12-020; classtype:attempted-admin; sid:21570; rev:8;)


# Detecting ms08-067
alert tcp any any -> any any (msg:"MS08-067 EXPLOIT DETECTED!!!"; flow:to_server,established; dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188; dce_opnum:31,32;
dce_stub_data; pcre:"/^(\x00\x00\x00\x00|.{4}(\x00\x00\x00\x00|.{12}))/s"; byte_jump:4,-4,multiplier 2,relative,align,dce; pcre:"/\x00\.\x00\.\x00[\x2f\x5c]/R";
metadata:policy balanced-ips drop, policy connectivity-ips drop, policy max-detect-ips drop, policy security-ips drop, service netbios-ssn; reference:cve,2008-4250;
reference:url,technet.microsoft.com/en-us/security/bulletin/MS08-067; classtype:attempted-admin; sid:14782; rev:21;)
```

And the result:



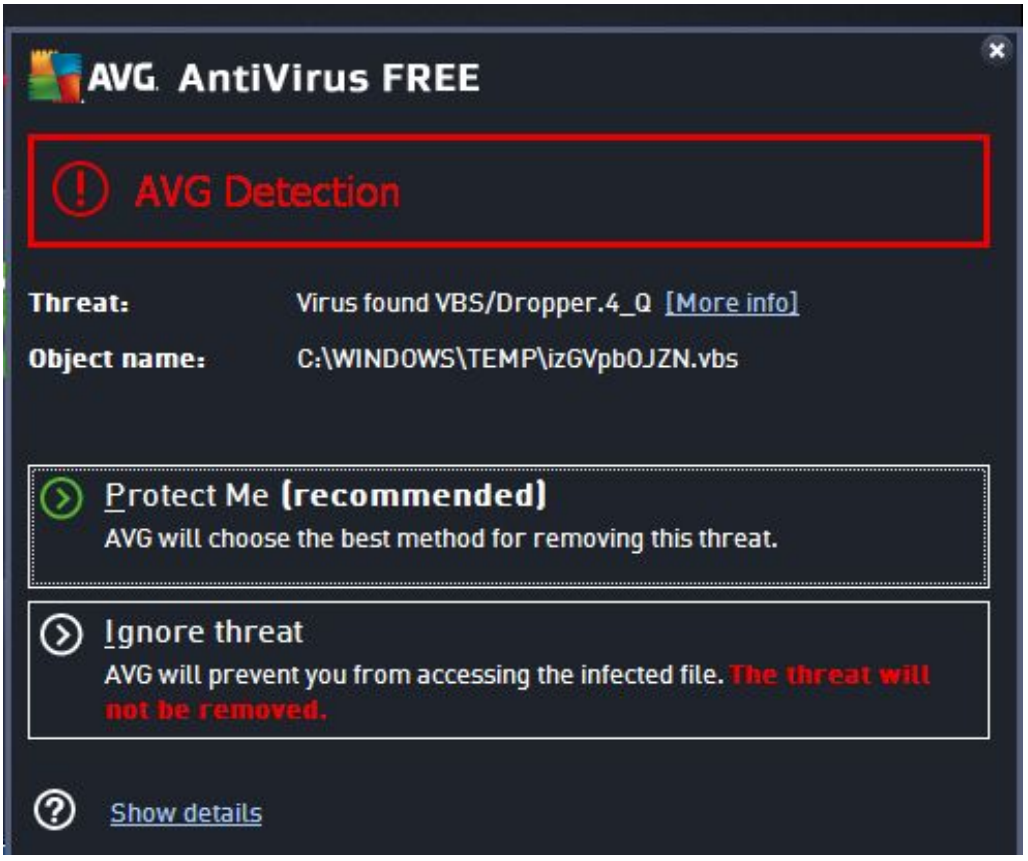
3.3.4 Installing Host Based Protections (10)

Antivirus

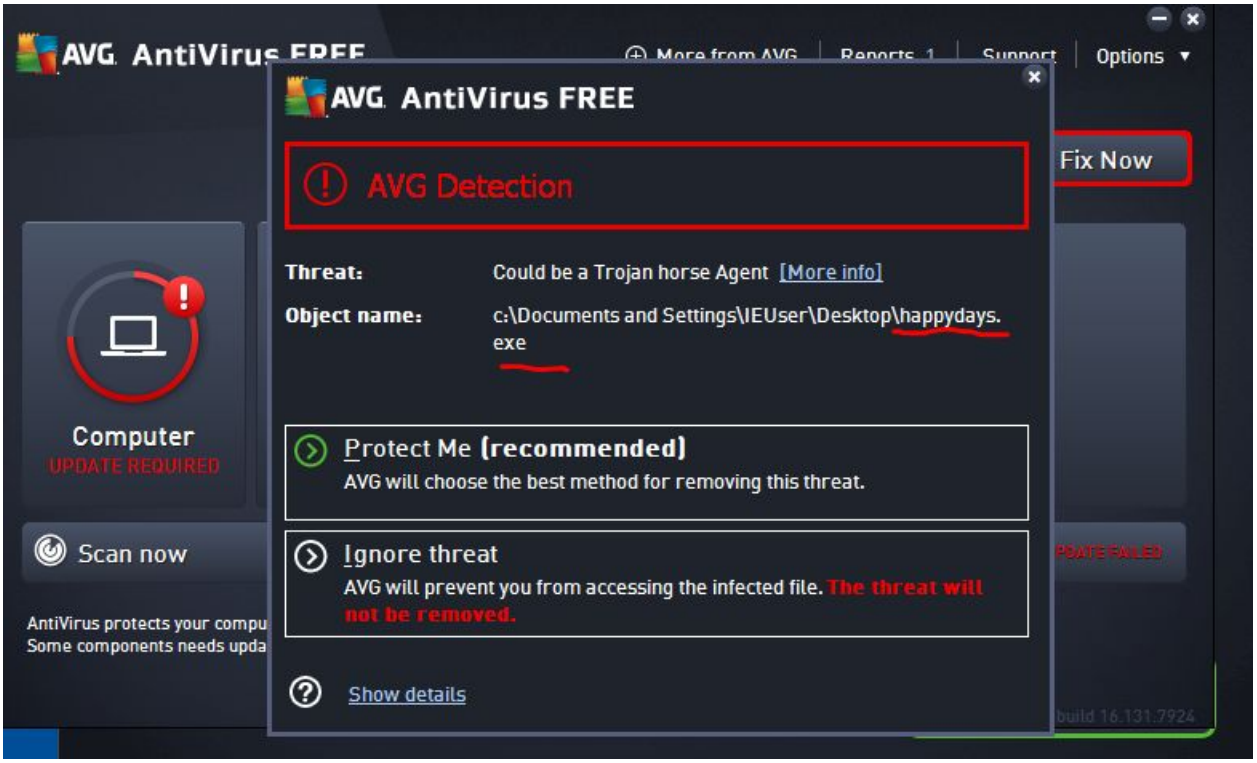


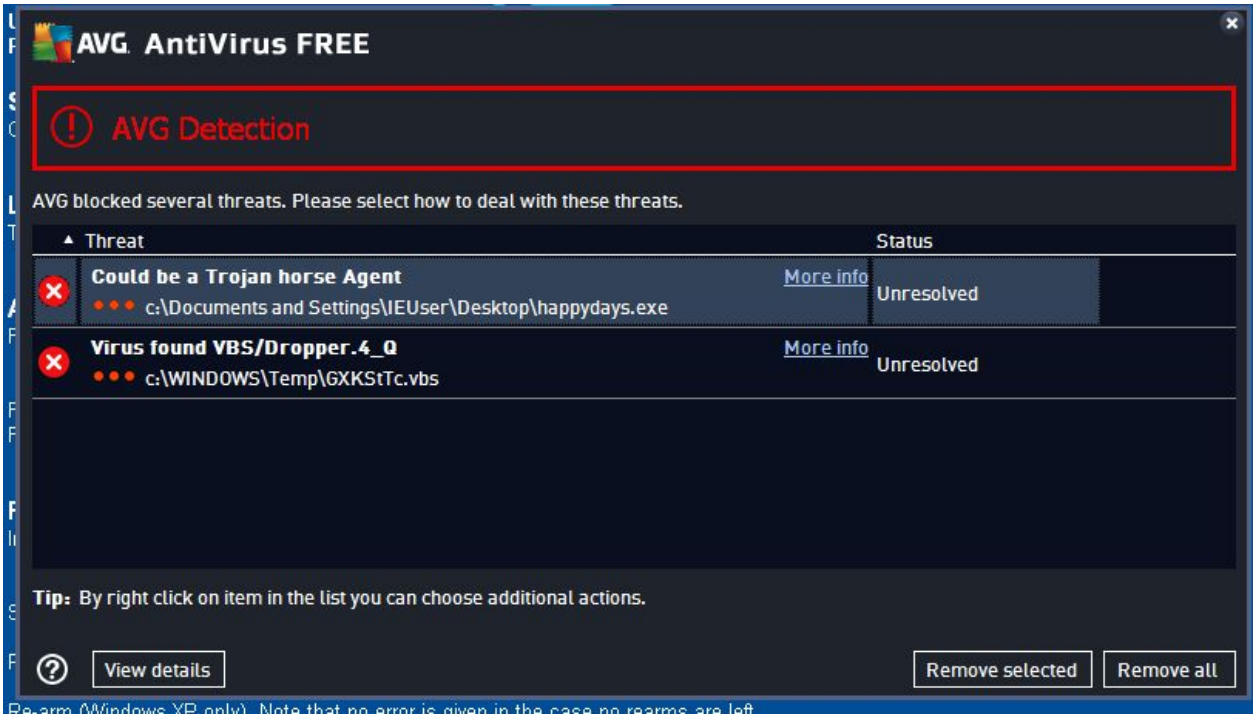
Installing and running AVG on the XP machine detected the above threat ('General behavioral detection' in a randomly named Windows system file). Lewis suspected that it was from the reverse listener exploit we had used to gain persistence.
We confirmed that selecting 'Protect Me' removed the persistence. We no longer got a shell when the XP machine booted up.

Regaining access and attempting to repeat the persistence exploit caused AVG to immediately recognise the offending file again.



AVG also detected the `happydays.exe` we had placed on the desktop.



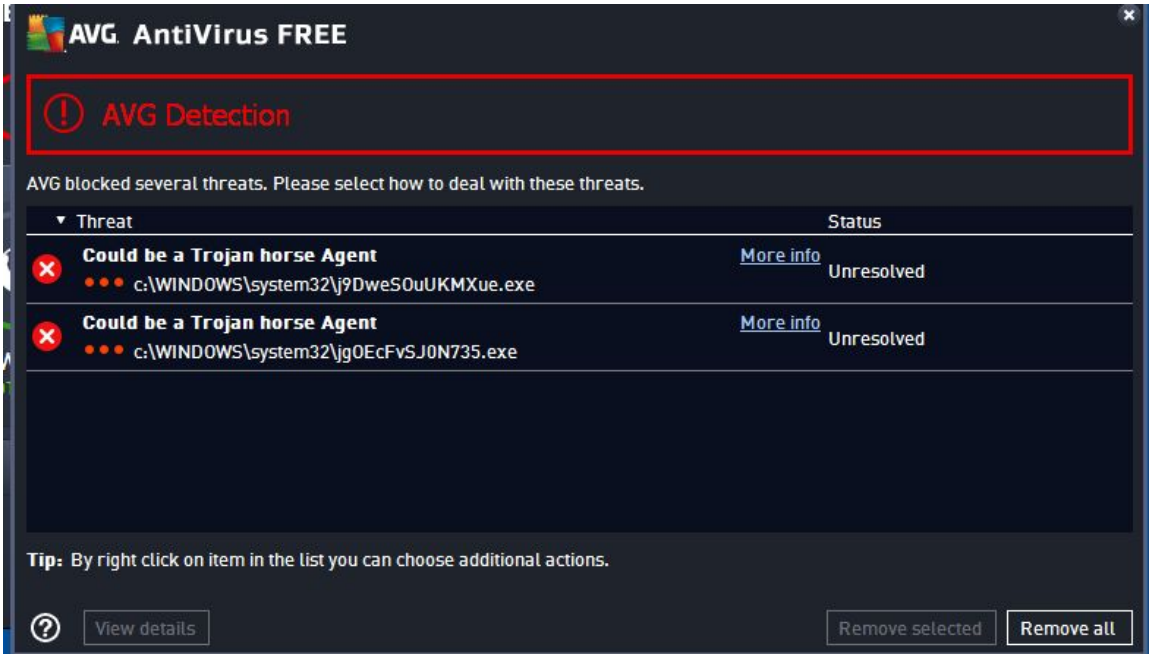


Rerunning Exploits

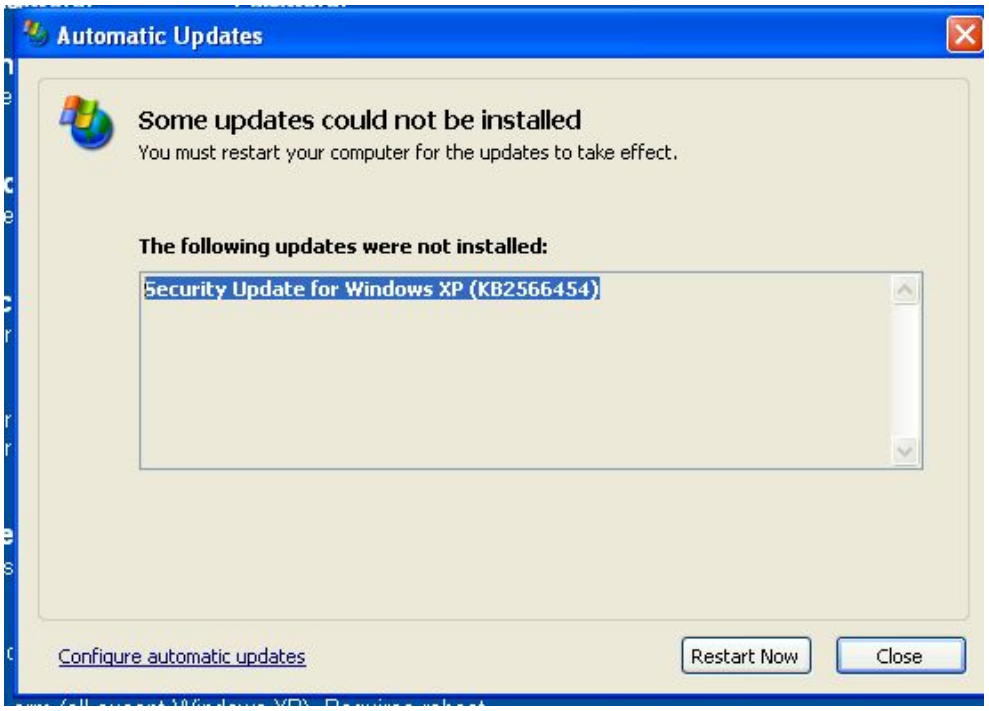
With AVG active, we reran the exploits we used in 3.2.5.

Microsoft Print Spooler Impersonation	Prevented
Microsoft Remote Desktop Protocol Code Execution	Worked
Internet Explorer Exploits	Worked
External Blue	Worked
Java Atomic Reference Array	Worked

When the print spooler impersonation was tried, AVG did the following.



Updating Windows



After updating windows, the remote desktop exploit (MS12-20) no longer worked.

3.3.5 Discussion of the Pros and Cons of Snort (10)

Pros

- Snort can be configured to look for various kinds of suspicious activity. We also read https://www.symantec.com/connect/articles/running-snort-part-2 for an overview of how snort can be used to detect intrusions that are not known exploits. Our main takeaways are:
    - Snort is configured via its ruleset to detect patterns of traffic that match known attacks (which is why the ruleset is downloaded separately from Snort itself, since it needs to be up to date). Clearly this is good for known attacks.
    - The user could configure snort to raise alerts when requests are received on the network for folders that are known to be sensitive and that remote users are not supposed to be accessing.
    - For files that are sensitive such as proprietary source code, a specific comment can be included in the text of the file (e.g. <!--IDS_SOURCE_CODE_TAG> in the Symantec article). Snort can be configured to notice that text when the file is transferred over the network, so an alert can be raised.
    - Snort can also be configured to look for incoming requests to run services that remote users should have no business running, such as `cmd.exe` or other services.
- Snort can be modified with custom rules at any time if there is a particular threat you are interested in watching for.
- Our investigation showed that Snort can be used to block a range of different kinds of traffic, if put inline so it can be configured as an IPS. So any of the above things could be set as rules to block rather than just detect those things, such as sensitive files being transferred or requests for remote root access to the system.
  https://www.darkreading.com/risk/squashing-malware-with-snort-in-line/d/d-id/1132079
- Snort is free.

Cons

- Snort's downloaded rule set contains rules to detect a range of known attack patterns. While good for known attacks, it is less good for new attacks or attacks that are not in the ruleset. The extra strategies taken from the Symantec article above (e.g. watching for requests to access known sensitive folders) give some possible mitigations to this problem.
- We found Snort very hard to set up and use. While clearly very powerful and potentially very important for detection and prevention, it has an ultra steep learning curve. Alister found that when trying to set it up on the XP machine that the paths in the snort.conf file were all incorrect for the windows version, and the links to the DLLs were also incorrect, despite having installed from the latest windows installer.
- On a busy network, Snort could potentially generate a large number of alerts, to the point that it could be difficult dealing with them all. One solution is for those that are certainly malicious to be straight up blocked rather than simply logged or alerted.
- Presumably Snort is capable of generating false positives as regards to suspicious vs legitimate traffic. Simply telling it to block everything it considered suspicious could lead to frustration by users. We believe setting up blocks that are useful but not restrictive could require the administrator to have in-depth knowledge of the specifics of the traffic on the network.
- The most up to date version of Snort's rules is not free and requires a monthly subscription.

3.3.6 Discussion of Other Countermeasures and Mitigations

We considered these additional countermeasures and mitigations to the Windows OS:

- **Firewall**: The obvious first line of defense. Checking around the net, we see that there are many kinds of firewalls with different capabilities. The basic implementation is to filter network traffic so only trusted information gets through, but using a more sophisticated firewall could allow you to detect more subtle patterns of malicious behavior. We note that there are many third party firewalls that are provided by antivirus companies, but there is also the built in windows firewall. Obviously, a sane person would never turn this off.
- **Updates**: In researching the various exploits to the XP machine, we have read a number of Microsoft Security Bulletins (e.g. https://docs.microsoft.com/en-us/security-updates/securitybulletins/2012/ms12-020) which illustrate to us the importance of keeping the operating system up to date. By following the dates of known version releases to the dates that they are patched out, we estimate that in the early 2000's most of the vulnerabilities lasted less than a month from when they were discovered. We believe this time is much shorter now. We understand that even if the vulnerability has been corrected by a patch, your system will remain vulnerable for as long as you haven't installed the patch.
- **Disable Unneeded Services**: Some of the exploits target specific services such as IRC or Telnet that many users don't need. For most people they won't be running those services by default, but it could pay to check and disable things that aren't needed in order to present a smaller attack surface. Based on the exploits we used against the VM, Windows Remote Desktop is a prime candidate as the exploits against it could cause catastrophic damage to the victim machine, and most people don't need it anyway. This could also include things like Adobe Flash, which has been known to have a number of security problems. If it is not needed, don't use it.
- **Antivirus**: By scanning frequently, known threats that have made their way onto the system can be identified. This clearly means keeping the virus scanner up to date. These days Windows does this for us, but as Windows XP is out of date and no longer supported, it does not do this for itself. In order to keep the system as secure as possible, we recommend a third party antivirus scanner such as Avast.
- **Don't Use Windows XP**: Clearly XP's primary vulnerability is that it is out of date and no longer maintained. The most obvious countermeasure against attacks is to not use XP at all as an operating system, as any existing and newly discovered vulnerabilities are not going to be addressed by Microsoft and can only be manually handled by software such as Snort or a third party firewall.