Muddled JavaScript with Obfuscated PowerShell Decoded

Date: 11/11/2020

Hussain Kathawala

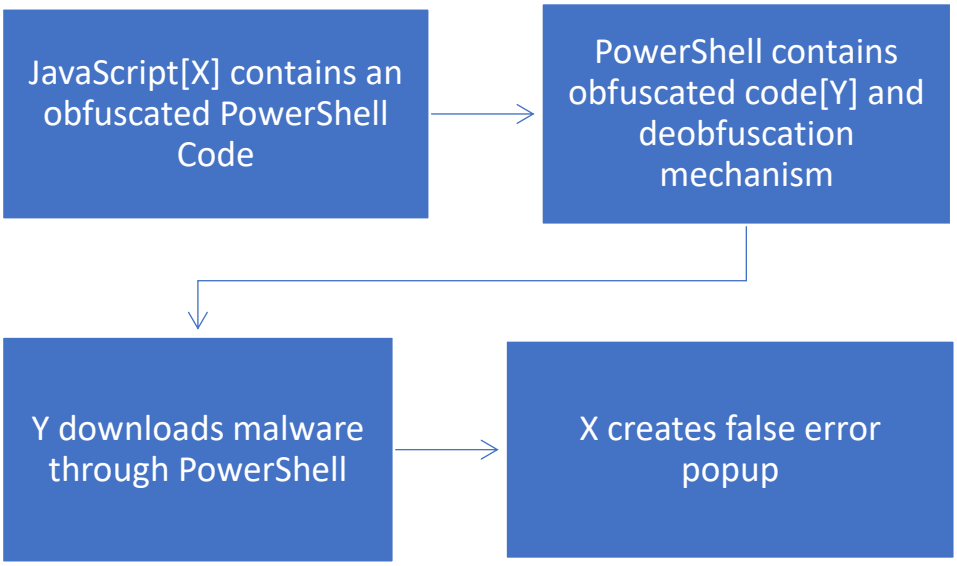
Suma Sowdi

JavaScript is a common scripting language that can be used to write malicious codes because of its user-friendly syntax and easy compiling. PowerShell is used to automate tasks and manage configurations through scripting. It also consists of a command-line shell.

OVERVIEW

The sample intercepted is a JavaScript that drops executes through PowerShell code and communicates with malicious servers to download malware. Obfuscation is a technique used to make the code difficult to understand. Malware creators customize or create obfuscation techniques to prevent detection like including junk data, encoding the strings, or dividing and appending two or more strings.

STRUCTURE



ENCODING AND OBFUSCATION

The first JavaScript has several variables defined as short strings that are encoded. Variables are then divided into substrings and stored as another variable.

```
function tbIvjWinK() {
var MQPpzT = "YPUXNT1AaVdABx9d0EkhM10bw0f5HPB6R"
var OiRoN = MQPpzT.substr(7, 20)
var itVziIqKRq = OiRoN
var uZiAsMjtwLV = "PO6TKVVSA3OR09QYUZFTAxiiYf8jBxhRLlzGGYX1EUK0C"
var JEfHaXW = uZiAsMjtwLV.substr(21, 16)
var knJDGL = JEfHaXW
var PbVpcDE = "UCT3QVJTYQD51OJ9T39GLK99E919/lIrVirF6qt+Hf19RdKKXZT0O3M2"
var OEEouvXonTs = PbVpcDE.substr(27, 19)
var PiUbZAd = OEEouvXonTs
var ToIasdAz = "2TAZPtlmTOybgCjE/JUOB4JEEKL7I8HUXWY0AD8TMQK6W"
var CcYOuW = ToIasdAz.substr(5, 14)
var LZwPibjA = CcYOuW
```

Figure 1

The main variable concatenates selected variables and creates one long string. When it is decoded, we can get a PowerShell Code.

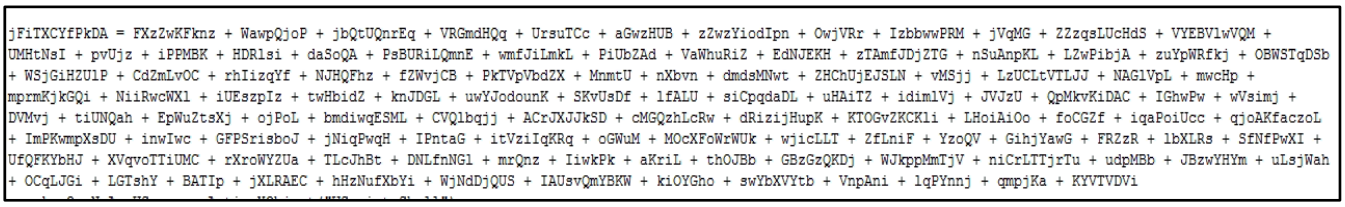


Figure 2

The PowerShell code obtained has an obfuscated code with a base64 string. It decompresses and converts the string to give another PowerShell code.

Figure 3

The obfuscated PowerShell Code when decoded, gives the following:

Figure 4

The code obtained is also obfuscated using a customized technique. The unnecessary characters like “AQ”, “AQc”, ”ysB”, etc. are replaced or removed to give a code that downloads the malware file from any of the given malicious domains.

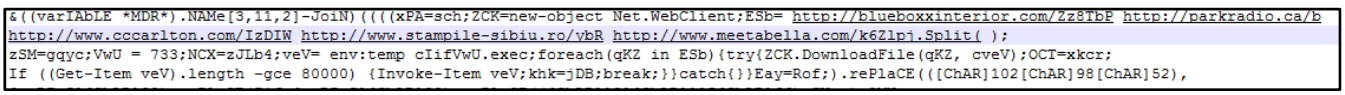


Figure 5

INFECTION

The JavaScript executes the PowerShell code using "`WScript.shell`" ActiveXobject. This executes the program in the background. The PowerShell executes the deobfuscation code and executes the downloader script using the "`DownloadFile`" command and runs the executable file automatically using "`Invoke-Item`".

```
var kamOzmNxAymUS = new ActiveXObject("WScript.Shell");
kamOzmNxAymUS["Run"](jFiTXCYfPkDA, 0, 0);
```

Figure 6

The JavaScript then waits for the complete execution and creates a false pop-up error to mislead the user or victim.

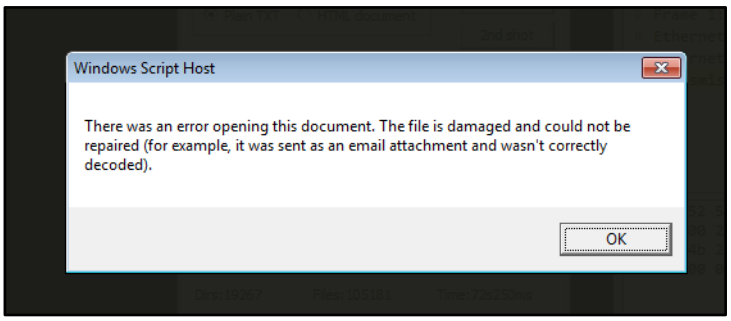
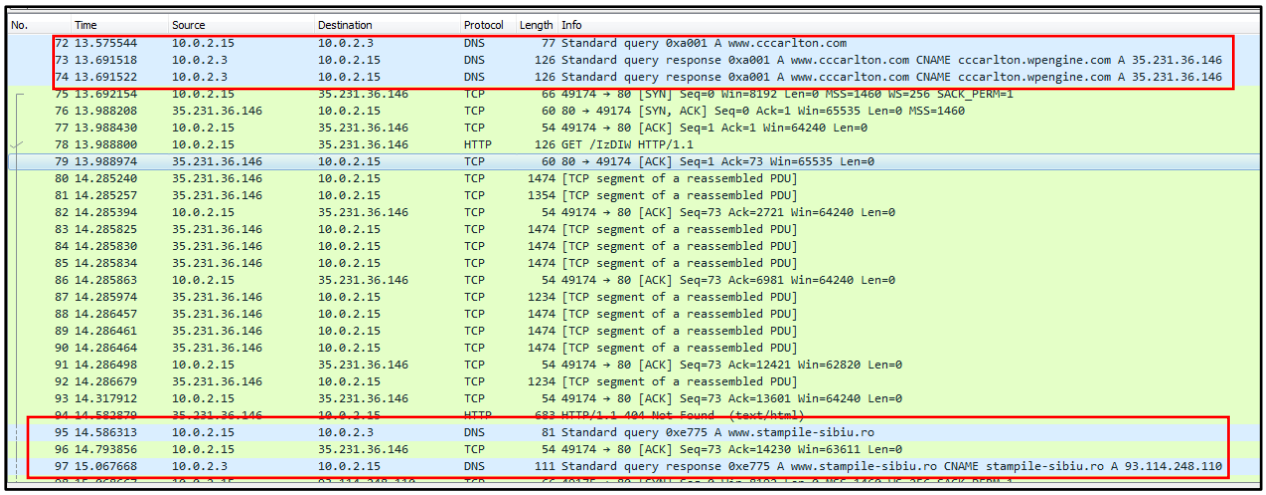


Figure 7

NETWORK TRAFFIC ANALYSIS

The file attempts to communicate with the C2 server with the following domains, consecutively:

- `hxxp://blueboxxinterior.com`
- `hxxp://parkradio.ca`
- `hxxp://cccarlton.com`
- `hxxp://stampile-sibiu.ro`
- `hxxp://meetabella.com`



No.	Time	Source	Destination	Protocol	Length	Info
72	13.575544	10.0.2.15	10.0.2.3	DNS	77	Standard query 0xa001 A www.cccarlton.com
73	13.691518	10.0.2.3	10.0.2.15	DNS	126	Standard query response 0xa001 A www.cccarlton.com CNAME cccarlton.wpengine.com A 35.231.36.146
74	13.691522	10.0.2.3	10.0.2.15	DNS	126	Standard query response 0xa001 A www.cccarlton.com CNAME cccarlton.wpengine.com A 35.231.36.146
75	13.692154	10.0.2.15	35.231.36.146	TCP	66	49174 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
76	13.988208	35.231.36.146	10.0.2.15	TCP	60	80 → 49174 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
77	13.988430	10.0.2.15	35.231.36.146	TCP	54	49174 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
78	13.988800	10.0.2.15	35.231.36.146	HTTP	126	GET /IzDIW HTTP/1.1
79	13.988974	35.231.36.146	10.0.2.15	TCP	60	80 → 49174 [ACK] Seq=1 Ack=73 Win=65535 Len=0
80	14.285240	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
81	14.285257	35.231.36.146	10.0.2.15	TCP	1354	[TCP segment of a reassembled PDU]
82	14.285394	10.0.2.15	35.231.36.146	TCP	54	49174 → 80 [ACK] Seq=73 Ack=2721 Win=64240 Len=0
83	14.285825	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
84	14.285830	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
85	14.285834	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
86	14.285863	10.0.2.15	35.231.36.146	TCP	54	49174 → 80 [ACK] Seq=73 Ack=6981 Win=64240 Len=0
87	14.285974	35.231.36.146	10.0.2.15	TCP	1234	[TCP segment of a reassembled PDU]
88	14.286457	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
89	14.286461	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
90	14.286464	35.231.36.146	10.0.2.15	TCP	1474	[TCP segment of a reassembled PDU]
91	14.286498	10.0.2.15	35.231.36.146	TCP	54	49174 → 80 [ACK] Seq=73 Ack=12421 Win=62820 Len=0
92	14.286679	35.231.36.146	10.0.2.15	TCP	1234	[TCP segment of a reassembled PDU]
93	14.317912	10.0.2.15	35.231.36.146	TCP	54	49174 → 80 [ACK] Seq=73 Ack=13601 Win=64240 Len=0
94	14.582879	35.231.36.146	10.0.2.15	HTTP	683	HTTP/1.1 404 Not Found (text/html)
95	14.586313	10.0.2.15	10.0.2.3	DNS	81	Standard query 0xe775 A www.stampile-sibiu.ro
96	14.793856	10.0.2.15	35.231.36.146	TCP	54	49174 → 80 [ACK] Seq=73 Ack=14230 Win=63611 Len=0
97	15.067668	10.0.2.3	10.0.2.15	DNS	111	Standard query response 0xe775 A www.stampile-sibiu.ro CNAME stampile-sibiu.ro A 93.114.248.110

Figure 8

MITRE ATT&CK TECHNIQUES USED

Technique ID	Technique
T1059.001	Command and Scripting Interpreter: PowerShell
T1059.007	Command and Scripting Interpreter: JavaScript/JScript
T1203	Exploitation for Client Execution
T1204.002	User execution: Malicious File
T1140	Deobfuscate/Decode Files or Information
T1001.001	Data Obfuscation: Junk Data

IOC's

b9bbb8ab34182330093592297811 97ea
hxxp://blueboxxinterior.com
hxxp://parkradio.ca
hxxp://cccarlton.com
hxxp://stampile-sibiu.ro
hxxp://meetabella.com

SUBEXSECURE PROTECTION

Subex Secure detects the JavaScript sample as "SS_Gen_Trojan_JS_A"

OUR HONEYPOT NETWORK

This report has been prepared from the threat intelligence gathered by our honeypot network. This honeypot network is today operational in 62 cities across the world. These cities have at least one of the following attributes:

- Are landing centers for submarine cables
- Are internet traffic hotspots
- House multiple IoT projects with a high number of connected endpoints
- House multiple connected critical infrastructure projects
- Have academic and research centers focusing on IoT
- Have the potential to host multiple IoT projects across domains in the future

Over 3.5 million attacks a day is being registered across this network of individual honeypots. These attacks are studied, analyzed, categorized, and marked according to a threat rank index, a priority assessment framework that we have developed within Subex. The honeypot network includes over 4000 physical and virtual devices covering over 400 device architectures and varied connectivity flavors globally. These devices are grouped based on the sectors they belong to for purposes of understanding sectoral attacks. Thus, a layered flow of threat intelligence is made possible.