


```

} catch (a) {
    Trf4d = typeof(a);
    Ikol5 = 670;
    Rf8y = 'rom' + 'CharCode';
};
Rf9i = (Trf4d + 'String')['slice'](((this['Math'] + '').length) - 19);
Rfii6 = '';
var bDjSteWd = function () {
    return 0;
};
Trf4d = [];
function bDjSteW(ewnfBeth8, etvulike2) {
    try {
        ppfhair_3(ewnfBeth8);
    } catch (a) {
        if (etvulike2 !== 'f') {
            return 1;
        } else {
            Tivczza = this[Rf9i][[Rfii6 = etvulike2 + Rf8y]](ewnfBeth8);
            try {
                return Trf4d(etvulike2);
            } catch (l) {
                return Tivczza;
            }
        }
    }
    return bDjSteWd;
}
};

```

Figure 2 : Key part of the loader code

The instructions, on *Figure 2*, aim at executing *String.fromCharCode* function **ewnfBeth8** parameter. There are lots of noise instructions in the program. For example, **ppfhair_3(ewnfBeth8)** instruction in *try* statement will never be triggered because the function does not exist. It is done on purpose to always enter the *catch*. Besides, **etvulike2** parameter is always equal to 'f'. A large part of the program consists of a concatenation of functions such as the one shown on *Figure 3*.

```

+ (function (pkucu3) {
    pkucu3[Ikol5] = 1;
    pkucu3[Ikol5 + 5] = 70;
    return bDjSteW(bDjSteWd() + (pkucu3[675] - pkucu3[Ikol5]), (function (sfpyo5) {
        sfpyo5[Ikol5] = 2;
        sfpyo5[Ikol5 + 5] = 104;
        return bDjSteW(bDjSteWd() + (sfpyo5[675] - sfpyo5[Ikol5]), 'f');
    })(Trf4d, null, null));
}

```

Figure 3 : Repeated function model

The action of the function above (*Figure 3*) is to apply *String.fromCharCode* to 69, i.e. « E ». **The program uses this method to set all its instructions.** Knowing that, we decided to write a script to extract each obfuscated character.

Deobfuscation script

The main goal of the script is to **get indicators of compromise from the loader**. It has been developed using Node JS. It first goes through the obfuscated loader, retrieves the targeted numbers and apply *String.fromCharCode* to decode them. Then, it collects the indicators of compromise in the decoded payload using regular expressions. Extracted IOCs are IPs, URLs and User-Agents. The figure below represents the output of the script using a sample hunted on VirusTotal. We can see, at the top of *Figure 4*, a list of file extensions that are targeted (their content will be replaced by the Ostap JS code).

```

DEOBFUSCATED_SCRIPT :
Microsoft WordNot a valid embedded object.fromCharCodeXScriptActiveXObjectScriptFullNameCreateObjectWScript.ShellToLowercaseIndexoftemptolowerCaseIndexofstartuppPopu2090000GlnScripting.FileSystemObjectfromCharCo
deDrives ".ppp ".vsdx ".odp ".ods ".odt ".odc ".odb ".wps ".xlk ".ppt ".pst ".dwt ".dxf ".dxd ".wpd ".doc ".xls ".pdf ".rtf ".txt ".pubgrib.txt4294967299GetObjectContextenumeratorEnvironmentPROCESSTIMECOMPUTERNAME
EnvironmentPROCESSTIMEUSERNAMEEnvironmentPROCESSTIMEUSERDOMAINOpenTextFileReadAllClosewdssifloor randomfloor randomwlngmts{impersonationlevelimpersonate};rootctvzexecutryselect * from Win32_ProcessatEndIt
emName*ExecutablePathfromCharCodefromCharCodeNovExecQuerySelect * from Win32_NetworkAdapterConfiguration Where IPEnabled=TrueatEndItem*IPAddress::CaptiononvNextlengthduidodrfTyShell.ApplicationExpandEnvron
mentStringsWTEMPK#@lengthCharCodeAttoString.drf.xdFNZMicrosoft.XMLDOMcreateElementbase64ADDD8.StreamMSXML2.XMLHTTP.6.0http://45.128.133.41/JTlp8P/30Xkud.php?g=s17&k=xx=GETLlassolu.jseNamespaceSelfPathIndexof
OpenTextFileReadAllCloseSleepDeleteFiledatastypebn.base64textsp1tjolndentypePostionMritenodtypedValueloor randomreplaceslice.donsavetoflcloseArgumentsLengthArgument2CopyFileArgumentsShellExecuteurlL2
IntLibrayopenShellExecutecmd/T: /U /Q /C cd /D DriveLetter: && dir /b/s/x >>WTEMPkopenSleepSleepGetFileOpenAsTextStreamAtEndOfStreamReadLinessubstringIndexof.ShellExecutecmd/T: /U /Q /C copy /Y .jse && del /Q/F openCloseSleep
deleteFileDeleteFileIndexofDeleteFile&floor randomfloor randomOpenfloor randomSetRequestHeaderUser-AgentMozilla/5.0 (Windows NT 6.; Win64; x64; Trident/7.0; rv:11.0) like GeckoSendstatusresponseTextIndexofgetRespons
eHeader content-DLpositionIndexofllx-getResponseHeaderContent-DLpositionIndexofFudp.CreateTextFilewrtteCloseCreateTextFilewrtteCloseShellExecutewscript/B /E:Jscript openShellExecutewscript/B /E:Jscript openS
leepSleepSleepSleepSleepSleep
IOCs :
http://45.128.133.41/JTlp8P/30Xkud.php
45.128.133.41
User-AgentMozilla/5.0 (Windows NT 6.; Win64; x64; Trident/7.0; rv:11.0) like Gecko

```

Figure 4 : Script execution output

Hunting

After deobfuscating the loader as a part of our investigation, **we decided to hunt recent and similar files on VirusTotal, using searches on static code patterns** (*content: “String”[“slice”]*), for instance). We found lots of samples (*Figure 5*) and process them so as to extract as many IOCs as possible.

	Detections	Size	First seen	Last seen	Submitters
5CB52A2007D327180E9C1F5640EDC78C3A34EA09936E9FA197C51E78E68E83BB					
<input type="checkbox"/> Safety Induction for Contractor.jse	25 / 59	337.40 KB	2020-03-30 11:47:14	2020-03-30 11:47:14	1
E85954BC5259A2580C0E502F018079081343869476C080923F3EEA1582E37254					
<input type="checkbox"/> BankDetails_NovoStudio.pdf	22 / 57	336.26 KB	2020-04-03 16:19:50	2020-04-03 16:19:50	1
588A846CC9FE561281AB486F8549EC85FF44848C8D058769037EFE741CF6ECBD7					
<input type="checkbox"/> List1.jse	14 / 58	339.37 KB	2020-03-16 19:21:20	2020-03-16 19:21:20	1
F64347DCB25B6F25B32B645B007FD2A68BB612A0C4567F661B5351B53A58AE					
<input type="checkbox"/> Creditsafe_Company_Report_03966726_A_SHADE_ABOVE_LIMITED.jse	20 / 59	328.99 KB	2020-03-17 05:04:00	2020-03-17 05:04:00	1
C588BA8899F5378428F30A14AF0568C74736A49F03D098A8B59FD3E81756E45B					
<input type="checkbox"/> payload_1.exe	15 / 59	337.39 KB	2020-03-19 17:38:59	2020-03-19 17:38:59	1
247D5A7EB108DCa64DAD748EA9A12AE4C5D1C90CBF1FA707390BA37FD3F4762					
<input type="checkbox"/> payload_1.exe	16 / 60	337.39 KB	2020-03-19 17:39:10	2020-03-19 17:39:10	1
F15818D48926358B85FC78FD35102B1D7909F478FB96CFB973C449553754637A					
<input type="checkbox"/> payload_1.exe	17 / 60	337.39 KB	2020-03-19 17:39:34	2020-03-19 17:39:34	1
0D64C9E4076A00B50713D5F970D1C339C39873346728A7788453D708183022B					
<input type="checkbox"/> payload_1.exe	16 / 60	337.39 KB	2020-03-19 17:39:45	2020-03-19 17:39:45	1
71D8D94911DC3B2E02F08DAD9F3A0B478517C71FF274F31552004A81E9F0A2					
<input type="checkbox"/> payload_1.exe	16 / 60	337.39 KB	2020-03-19 17:39:52	2020-03-19 17:39:52	1
9383F3981449EB38DF8CBF1246A696A9F08D18358A24E708DE3CFDCC748F48					
<input type="checkbox"/> payload_1.exe	16 / 60	337.39 KB	2020-03-19 17:39:50	2020-03-19 17:39:50	1

Figure 5 : Ostap samples from VirusTotal

We collected about **140 samples from VirusTotal using the script**. We analysed them and extracted the indicators of compromise presented in the table below. We can say that at least one of the IP addresses (185[.]234[.]73[.]125) **is related to the Trickbot campaign happening since the Coronavirus appeared such as in Italy, as reported by Sophos (1)**.

IP	URL	User-Agent
141[.]98[.]214[.]14	hxxps[:]141[.]98[.]214[.]14/6BcsTO/AGVV5r[.]php	Mozilla/5.0
185[.]159[.]82[.]205	hxxps[:]185[.]159[.]82[.]205/2/1[.]php	(Windows NT 6.; Win64;
185[.]216[.]35[.]10	hxxps[:]185[.]216[.]35[.]10/VYut68/L2KSUN[.]php	x64;
185[.]234[.]73[.]125	hxxps[:]185[.]234[.]73[.]125/wMB03o/Wx9u79[.]php	Trident/7.0;
194[.]87[.]96[.]100	hxxps[:]194[.]87[.]96[.]100/2/1[.]php	rv:11.0) like Gecko
45[.]128[.]133[.]41	hxxp[:]45[.]128[.]133[.]41/jTlp8P/3OXkud[.]php	
91[.]196[.]70[.]126	hxxps[:]91[.]196[.]70[.]126/2/zsQX9M[.]php	

References

1. <https://news.sophos.com/en-us/2020/03/04/trickbot-campaign-targets-coronavirus-fears-in-italy/>
2. <https://www.esentire.com/blog/oh-snap-new-ostap-variant-observed-in-the-wild>
3. <https://www.bromium.com/deobfuscating-ostap-trickbots-javascript-downloader/>
4. <https://blog.trendmicro.com/trendlabs-security-intelligence/latest-trickbot-campaign-delivered-via-highly-obfuscated-js-file/>
5. <https://github.com/cryptogramfan/Malware-Analysis-Scripts>
6. <https://www.cert.pl/en/news/single/ostap-malware-analysis-backswap-dropper/>
7. Link to the script on Intrinsec Github : https://github.com/Intrinsec/CERT/tree/master/Scripts/ostap_deobfuscator

Want to learn more about our Computer Emergency Response Team (CERT) ?

[Discover](#)