More Than Meets the Eye: Exposing a Polyglot File That Delivers IcedID

By Mark Lim

Published: 2022-09-27 · Archived: 2026-04-06 01:51:15 UTC

Executive Summary

Unit 42 recently observed a polyglot Microsoft Compiled HTML Help (CHM) file being employed in the infection process used by the information stealer IcedID. We will show how to analyze the polyglot CHM file and the final payload so you can understand how the sample evades detection.

Multiple attack groups such as Starchy Taurus (aka APT41) and Evasive Serpens (formerly tracked as OilRig, also known as Europium) have abused CHM files to conceal payloads written using PowerShell or JavaScript. Here, we describe an interesting attack that allows attackers to avoid the need for long lines of code, which can make it easier for malicious files to evade detection by security products. Polyglot files can be abused by attackers to hide from anti-malware systems that rely on file format identification. The technique involves executing the same CHM file twice in the infection process. The first execution exhibits benign activities, while the second execution stealthily carries out malicious behaviors.

This particular attack chain was discovered in early August 2022 and delivered IcedID, also known as Bokbot, as the final payload. This information stealer, IcedID, is well-known malware that has been attacking users since 2019.

Palo Alto Networks customers receive protections from malware families using similar anti-analysis techniques with Cortex XDR or the Next-Generation Firewall with cloud-delivered security services including WildFire, Advanced Threat Prevention, Advanced URL Filtering and DNS Security.

Malicious Polyglot CHM File

Polyglot files are binaries that have multiple different file format types. The file would have a different behavior depending on the application that was used to execute it.

The attack that was discovered in early August 2022 starts with a phishing email that includes an attached zip file named erosstrucking-file-08.08.2022.zip. The zip file decompresses into an ISO image file named order-130722.28554.iso. Inside the ISO file is a CHM file called pss10r.chm (SHA256: 3d279aa8f56e468a014a916362540975958b9e9172d658eb57065a8a230632fa). The polyglot CHM file is used to display help documentation. When the user launches the CHM file (pss10r.chm), a harmless help window is displayed.

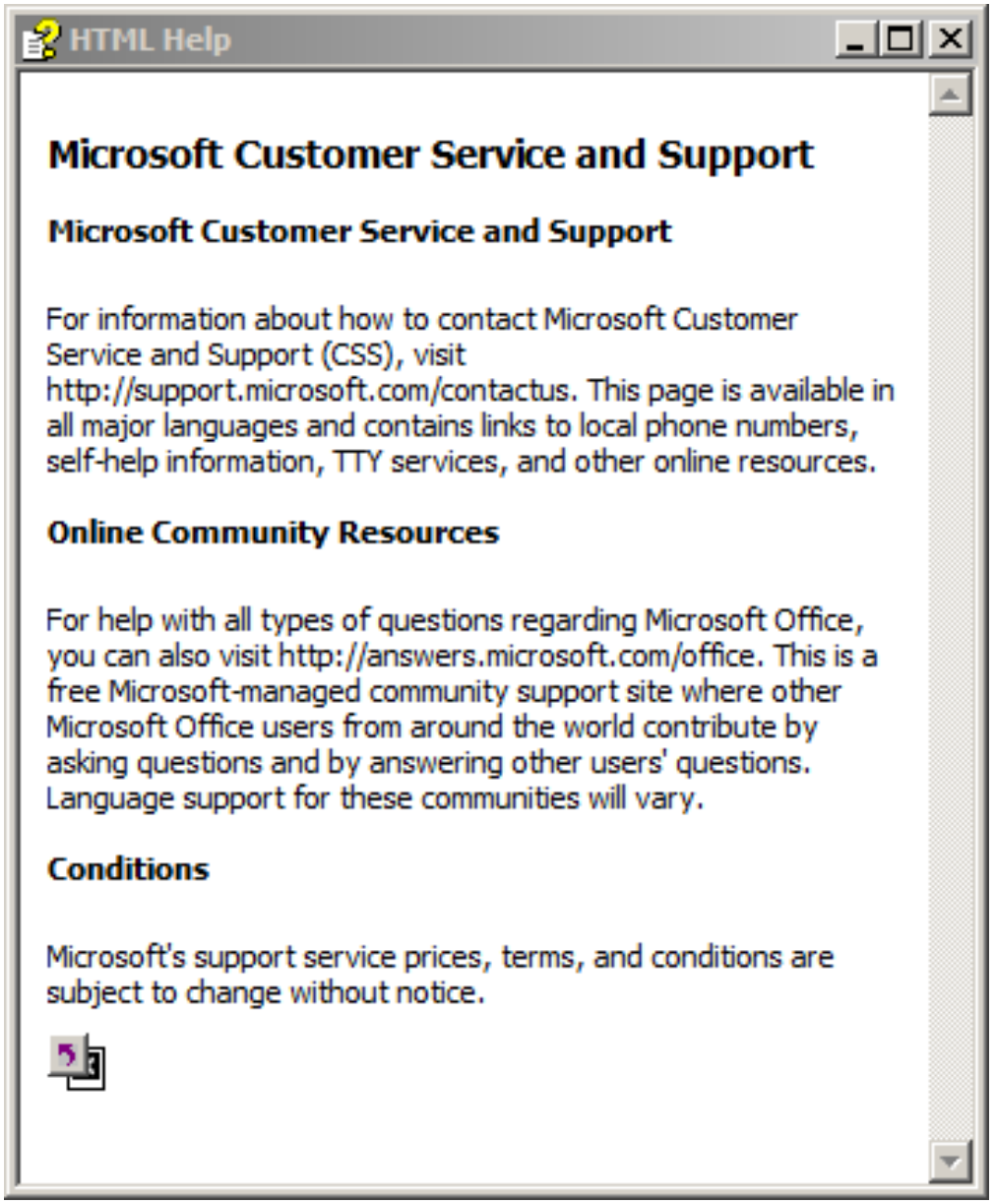

Figure 1. Decoy HTML help window.

To dump the contents of the CHM file, we used 7zip. The file of interest is PSSXMicrosoftSupportServices_HP05221271.htm.

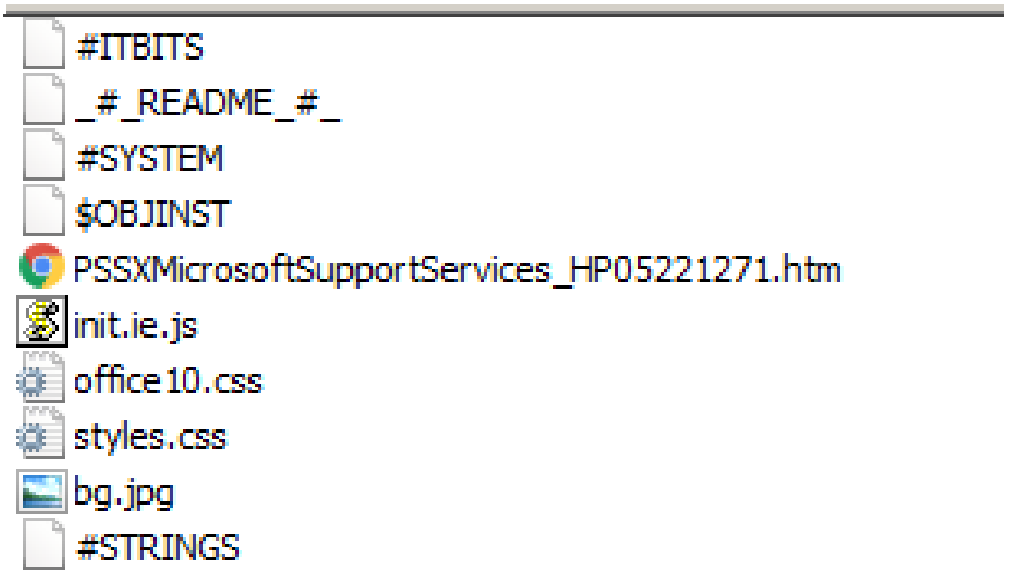

Figure 2. Contents of the decoy HTML help window.

Most of the code in the HTML file is used for generating the decoy window. However, concealed within the HTML code is a single-line command to execute the same CHM file again. The command calls Mshta.exe to execute itself (pss10r.chm) a second time. Mshta.exe is a utility that executes Microsoft HTML Application (HTA) files. HTAs are full-fledged applications created using HTML.

Figure 3. A line of HTML code in PSSXMicrosoftSupportServices_HP05221271.htm that calls Mshta.exe to execute the CHM file a second time.

The code of the HTA is buried within the binary of the CHM file and configured to be invisible to the victim during execution. The HTA is used to execute the binary app.dll.

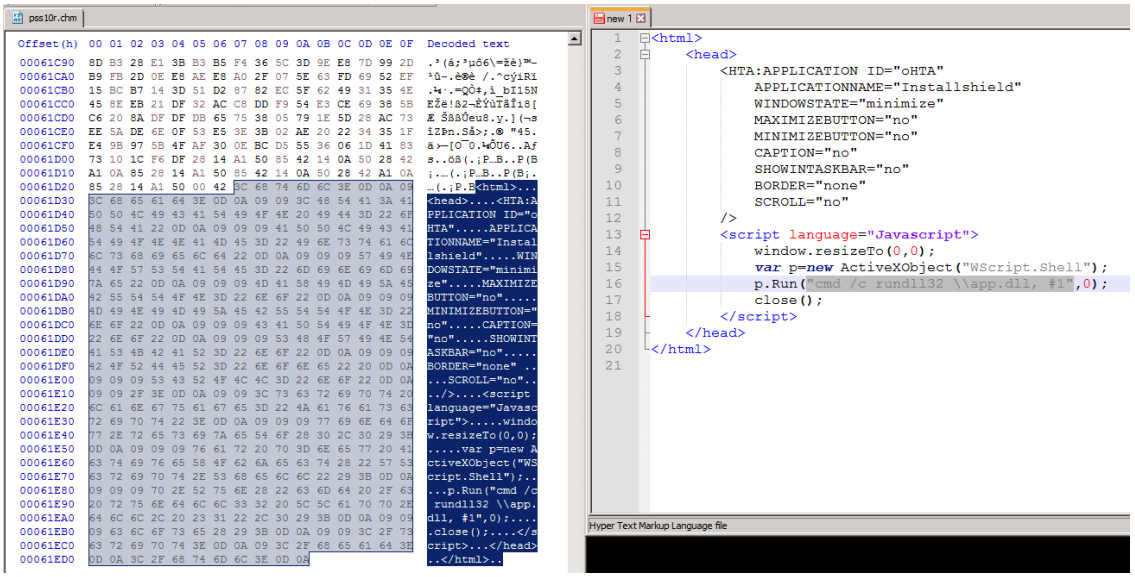

Figure 4. HTA code buried in pss10r.chm.

The binary app.dll is actually hidden within the ISO image. The hidden binary can be revealed using the attrib command.

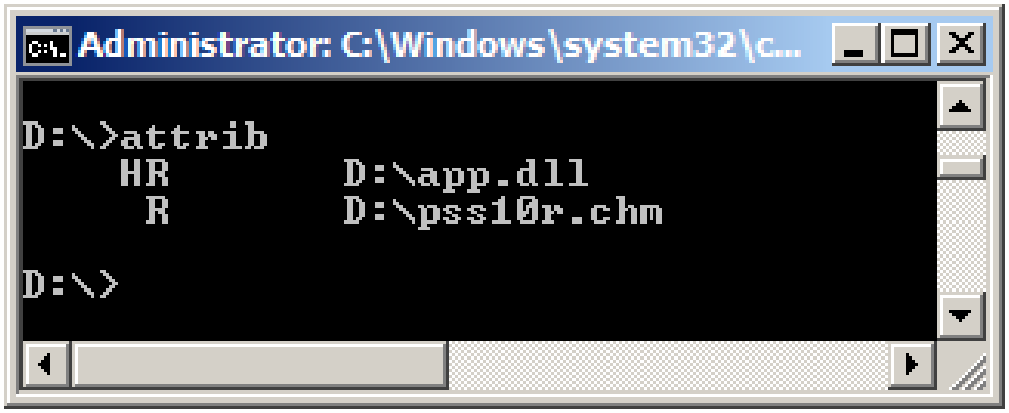

Figure 5. Revealing the hidden binary.

The *app.dll* binary is a 64-bit IcedID DLL. (SHA256: d240bd25a0516bf1a6f6b3f080b8d649ed2b116c145dd919f65c05d20fc73131)

IcedID DLL's Configuration Extraction

To retrieve the indicators of compromise (IoCs) from the IcedID DLL, we looked at its configuration. The IcedID DLL's configuration is encoded and stored in the data section of the binary. The encoded configuration has the format shown in Figure 6.

```
struct Encoded_config_blob
{
    Buffer [0x40] XOR_Key;
    Buffer [0x40] Enc_config;
}
```

Figure 6. Structure of encoded configuration blob.

The following function would decode the IcedID DLL's configuration at runtime. The address of the encoded configuration (enc_config) is in the function.

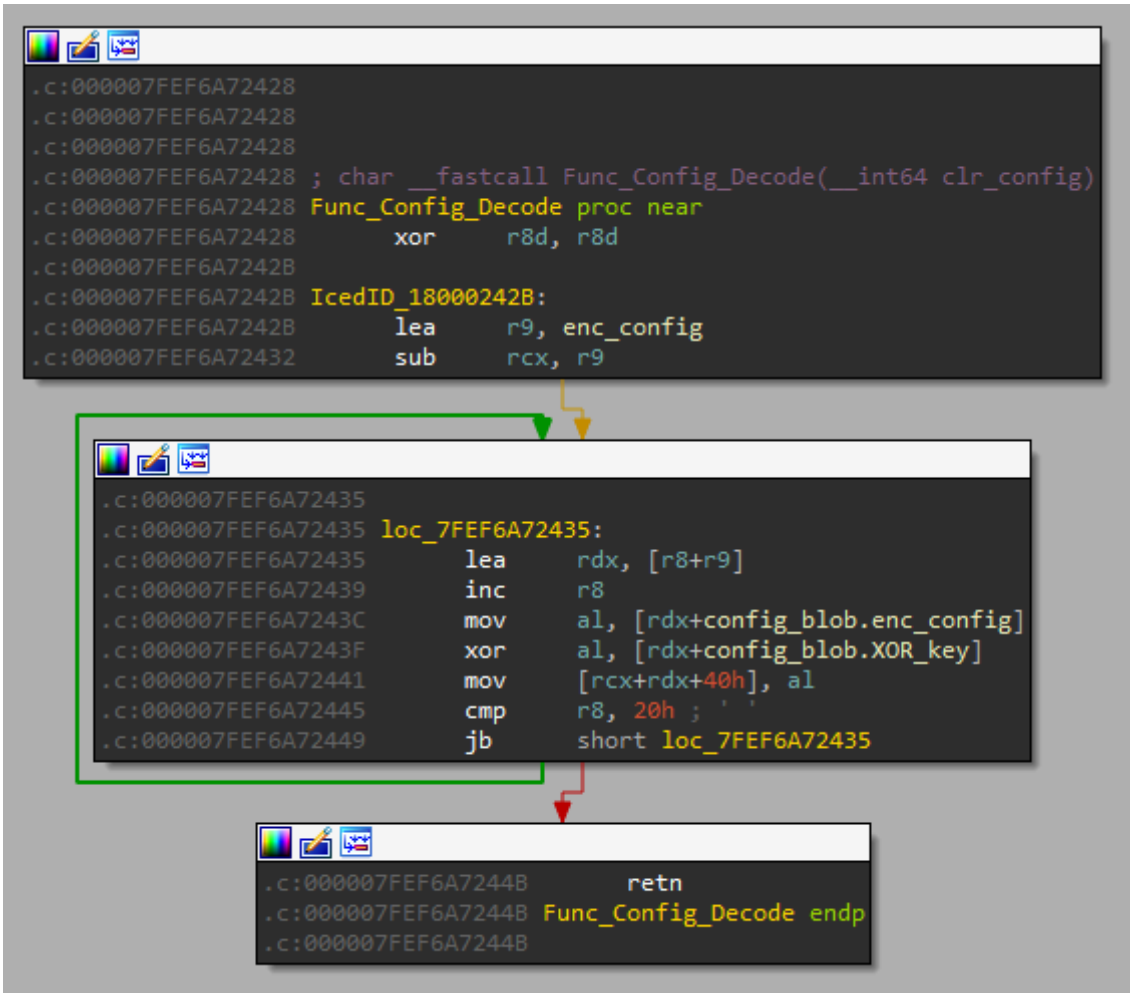

Figure 7. IcedID DLL's configuration decoder function.

The decoded IcedID DLL's configuration has the following format.

```
struct IcedID_config
{
    DWORD Campaign_ID;
    Buffer [0x1C] C2_URL;//C String
}
```

Figure 8. Structure of decoded IcedID configuration.

From the decoded configuration, we can extract the following IoCs:

Command and Control URL	abegelkunic[.]com
Campaign ID	4157420015

Indicators of Compromise

File name: erosstrucking-file-08.08.2022.zip
SHA256: fb6d23f69d14d474ce096da4dcfea27a84c93f42c96f6dd8295d33ef2845b6c7

File name: order-130722.28554.iso
SHA256: d403df3fb181560d6ebf4885b538c5af86e718fecfabc73219b64924d74dd0eb

File name: pss10r.chm
SHA256: 3d279aa8f56e468a014a916362540975958b9e9172d658eb57065a8a230632fa

File name: app.dll
SHA256: d240bd25a0516bf1a6f6b3f080b8d649ed2b116c145dd919f65c05d20fc73131

Command and Control URL: abegelkunic[.]com
