The First Stage of ShadowHammer

norfolkinfosec.com/the-first-stage-of-shadowhammer/

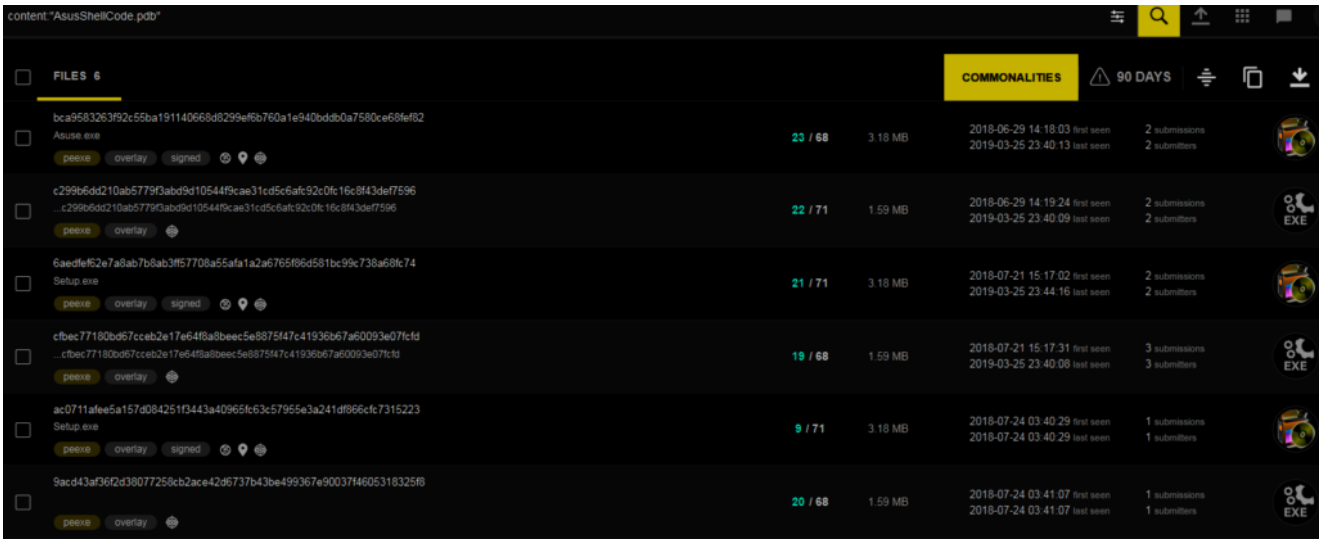
norfolk March 27, 2019

On 25 March, Kaspersky researchers published details of a supply chain compromise involving ASUS, a Taiwan-based computer manufacturer. As part of this compromise, a threat actor pushed malicious code to victims who connected to the company's servers using the ASUS Live Update feature used to deliver drivers and other updates (this blog notes that such update platforms are common across all manufacturers).

The malicious code in question is a first-stage triage tool, and details of the second-stage code have not yet been uncovered. This post documents this first-stage functionality of one of the identified variants, which compares the victim's MAC address to a hardcoded list prior to communicating with a C2.

Malware Workflow

Kaspersky provided a malicious hash in its original blog post; however, pivoting using content from a tweet from Costin Raiu yields additional files. Examination of these files (either through the VirusTotal platform or manually) indicates that they come in pairs: the smaller executables are hardcoded resources within the larger executables. A full hash list is available at the end of this post, and the hash used for this analysis is below.



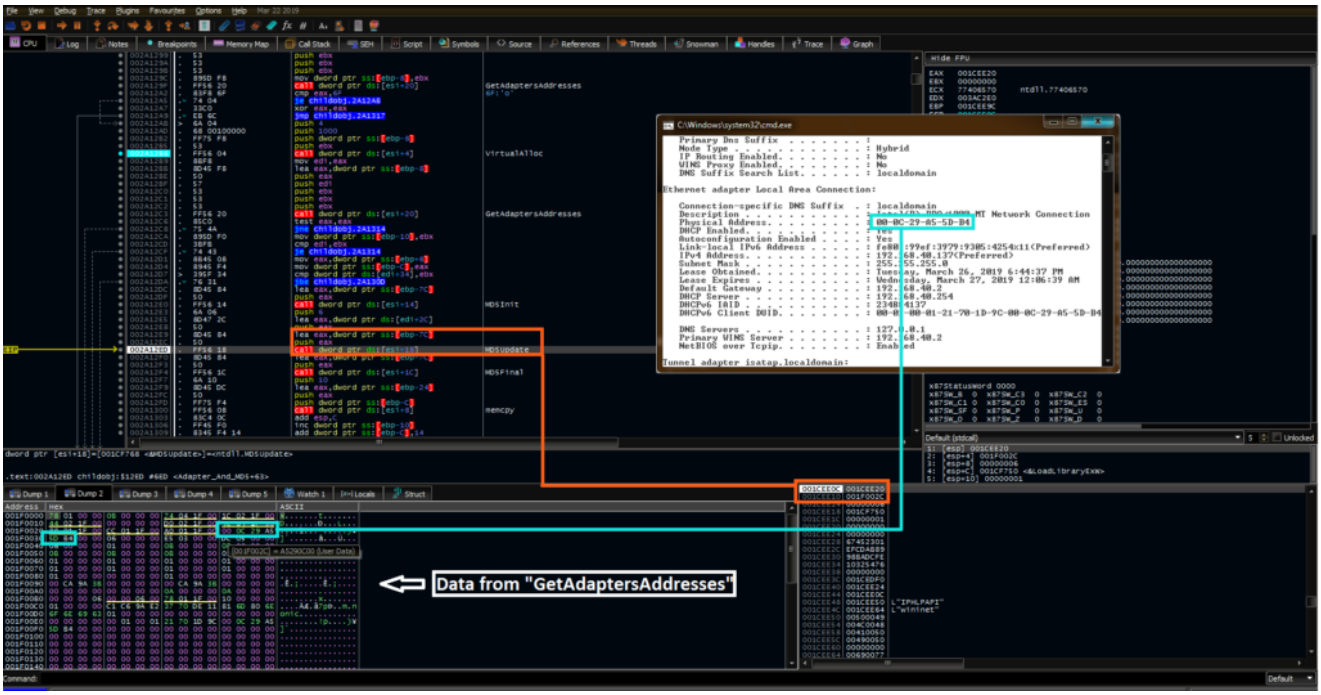
MD5: fa83ffde24f149f9f6d1d8bc05c0e023
SHA1: b0127ce307589ef48e2658784dd83ef7aa26097b
SHA256: c299b6dd210ab5779f3abd9d10544f9cae31cd5c6afc92c0fc16c8f43def7596

At the start of the malicious workflow for this code, the program dynamically resolves the LoadLibraryEx and GetProcAddress calls, which are in turn used to load additional APIs, including:

– memcpy (ntdll)
– memcmp (ntdll)
– memset (ntdll)
– MD5Init (ntdll)
– MD5Update (ntdll)
– MD5Final (ntdll)
– GetAdaptersAddresses (iphlpapi)
– InternetOpenA (wininit)
– InternetOpenURLA (wininit)
– InternetQueryDataAvailable (wininit)
– InternetReadFile (wininit)

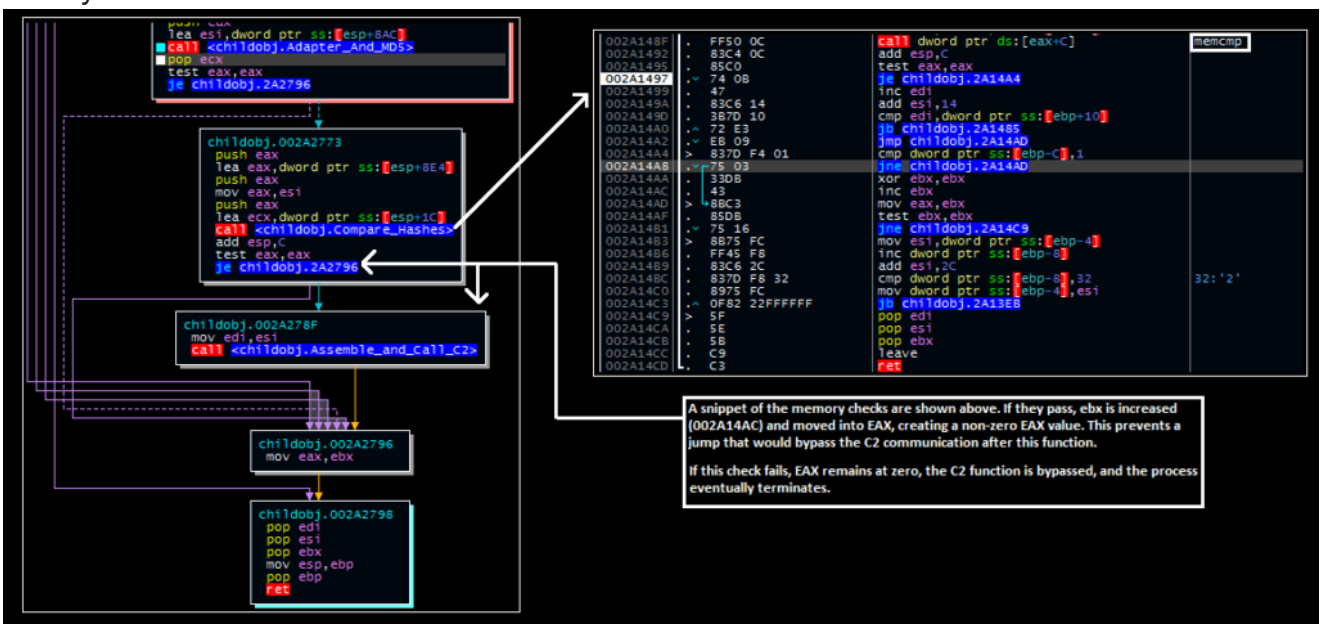
The malware then begins allocating large sections of code to memory and begins the triage process. The first function in this workflow queries the device's network adapters through the GetAdaptersAddresses API call resolved above. The network adapter information is allocated into memory. Using the MD5 Init, Update, and Final calls, the malware calculates an MD5 hash of each adapter's MAC address and stores this in memory for later comparison. Documentation for these functions can be found here.

This workflow is shown below- the orange represents the parameters for the MD5Update call, one of which is the address highlighted in blue of the stored MAC address.

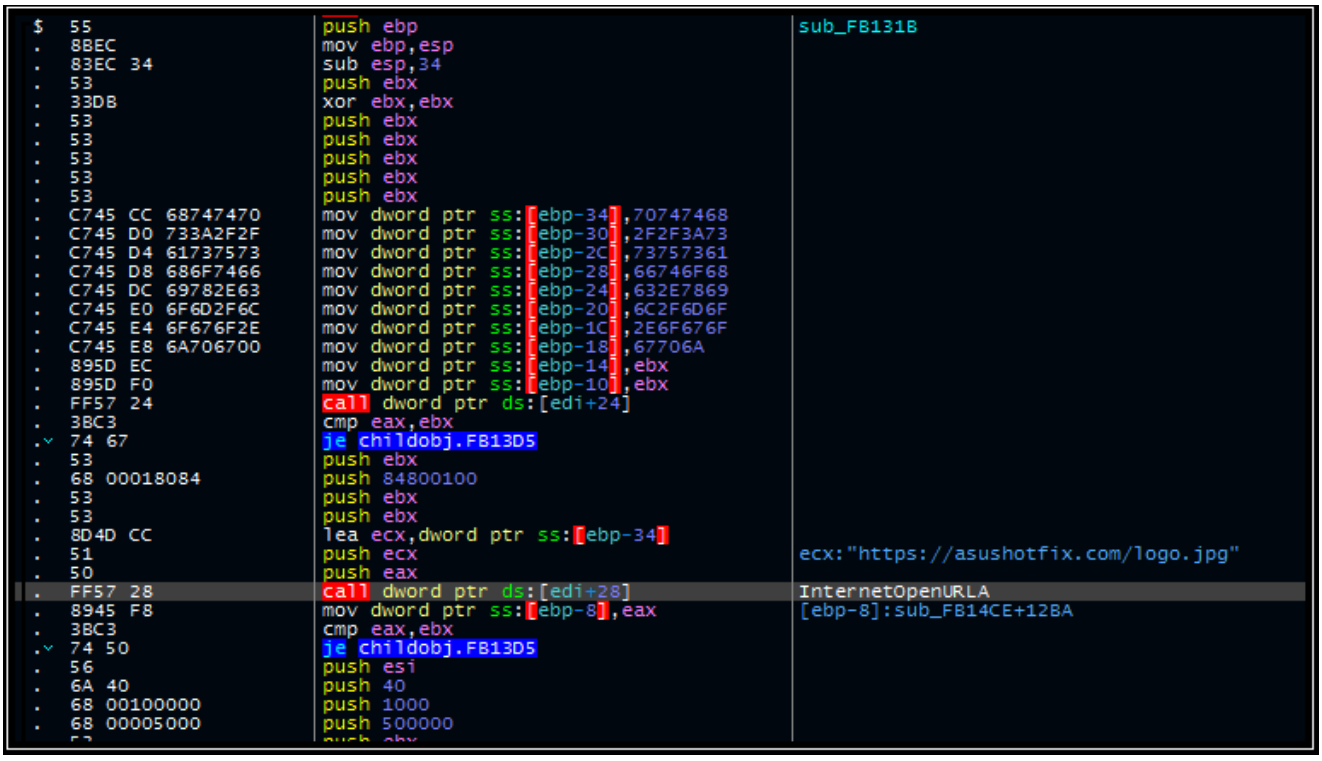


After these hash values are calculated and stored, the program compares them to a hardcoded set of MD5s. This step determines whether or not the malware will call out to its C2. By performing comparisons of the hashes of these values rather than the values themselves, the malware author increases the difficulty of identifying potential targets.

At a high level, if no match is found, the C2 routine is bypassed and the malware terminates shortly thereafter. A lower-level workflow for this is shown below:



Should a match be found, the “test eax eax” instruction will not set the zero flag, preventing the malware from jumping over the function and initiating the C2 call-out for the next (currently undisclosed) stage of activity:



Additional Indicators

C2
asushotfix[.]com

Hashes

MD5
17a36ac3e31f3a18936552aff2c80249
fa83ffde24f149f9f6d1d8bc05c0e023
f2f879989d967e03b9ea0938399464ab
5855ce7c4a3167f0e006310eb1c76313
2a95475af7a07ee95ab11caad9e99b0c
cb3f78d3ff776a7afe6c56371b0c7e11

SHA1
5039ff974a81caf331e24eea0f2b33579b00d854
b0127ce307589ef48e2658784dd83ef7aa26097b
2c591802d8741d6aef1a278b9aca06952f035b8f
0d9d48a4545120d84df6614378456ad722d82f58
0595e34841bb3562d2c30a1b22ebf20d31c3be86
ffdb4f49a96f382161907ea21146332d2defb7b5

SHA256
bca9583263f92c55ba191140668d8299ef6b760a1e940bddb0a7580ce68fef82
c299b6dd210ab5779f3abd9d10544f9cae31cd5c6afc92c0fc16c8f43def7596
6aedfef62e7a8ab7b8ab3ff57708a55afa1a2a6765f86d581bc99c738a68fc74
cfbec77180bd67cceb2e17e64f8a8beec5e8875f47c41936b67a60093e07fcfd
ac0711afee5a157d084251f3443a40965fc63c57955e3a241df866cfc7315223
9acd43af36f2d38077258cb2ace42d6737b43be499367e90037f4605318325f8