Winnti Group targeting universities in Hong Kong

By Mathieu Tartare

In November 2019, we discovered a new campaign run by the Winnti Group against two Hong Kong universities. We found a new variant of the ShadowPad backdoor, the group's flagship backdoor, deployed using a new launcher and embedding numerous modules. The Winnti malware was also found at these universities a few weeks prior to ShadowPad.

The Winnti Group, active since at least 2012, is responsible for for high-profile supply-chain attacks against the video game and software industries leading to the distribution of trojanized software (such as CCleaner, ASUS LiveUpdate and multiple video games) that is then used to compromise more victims. It is also known for having compromised various targets in the healthcare and education sectors.

ESET researchers recently published a white paper updating our understanding of the arsenal of the Winnti Group, following a blog post documenting a supply-chain attack targeting the videogame industry in Asia. Additionally, we published a blog post on a new backdoor named skip-2.0 that targets Microsoft SQL Server.

This article focuses on the technical details of this new ShadowPad variant.

About the "Winnti Group" naming:
We have chosen to keep the name "Winnti Group" since it's the name first used to identify it, in 2013, by Kaspersky. Since Winnti is also a malware family, we always write "Winnti Group" when we refer to the malefactors behind the attacks. Since 2013, it has been demonstrated that Winnti is only one of the many malware families used by the Winnti Group.

ShadowPad found at several Hong Kong universities

In November 2019, ESET's machine-learning engine, Augur, detected a malicious and unique sample present on multiple computers belonging to two Hong Kong universities where the Winnti malware had already been found at the end of October. The suspicious sample detected by Augur is actually a new 32-bit ShadowPad launcher. Samples from both ShadowPad and Winnti found at these universities contain campaign identifiers and C&C URLs with the names of the universities, which indicates a targeted attack.

In addition to the two compromised universities, thanks to the C&C URL format used by the attackers we have reasons to think that at least three additional Hong Kong universities may have been compromised using these same ShadowPad and Winnti variants.

This campaign of the Winnti Group against Hong Kong universities was taking place in the context of Hong Kong facing civic protests that started in June 2019 triggered by an extradition bill. Even though the bill was withdrawn in October 2019, protests continued, demanding full democracy and investigation of the Hong Kong police. These

protests gathered hundreds of thousands of people in the streets with large support from students of Hong Kong universities, leading to multiple university campus occupations by the protesters.

We have contacted the compromised universities and provided the necessary information and assistance to remediate the compromise.

Updated launcher

Unlike previous ShadowPad variants documented in our white paper on the arsenal of the Winnti Group, this launcher is not obfuscated using VMProtect. Furthermore, the encrypted payload is neither embedded in the overlay nor located in a COM1:NULL.dat alternate data stream. And the usual RC5 encryption with a key derived from the volume ID of the system drive of the victim machine (as seen in the PortReuse backdoor, skip-2.0 and some ShadowPad variants) is not present either. In this case, the launcher is much simpler.

DLL side-loading

The launcher is a 32-bit DLL named hpqhvsei.dll, which is the name of a legitimate DLL loaded by hpqhvind.exe. This executable is from HP and is usually installed with their printing and scanning software called “HP Digital Imaging”. In this case the legitimate hpqhvind.exe was dropped by the attackers, along with their malicious hpqhvsei.dll, in C:\Windows\Temp.

Although we do not have the component that dropped and executed this launcher, the presence of these files leads us to think that the initial execution of this launcher is done through DLL side-loading.

When the malicious DLL is loaded at hpqhvind.exe startup, its DLLMain function is called that will check its parent process for the following sequence of bytes at offset 0x10BA:

85 C0 ; test eax, eax
0F 84 ; jz

In the case where the parent process is hpqhvind.exe, this sequence of bytes is present at this exact location and the malicious DLL will proceed to patch the parent process in memory. It replaces the original instructions at 0x10BA with an unconditional jump (jmp – 0xE9) to the address of the function from hpqhvsei.dll that decrypts and executes the encrypted payload embedded in the launcher.

The decompiled function responsible for patching the parent process is shown in Figure 1. In case hpqhvsei.dll is loaded by a different process than hpqhvind.exe, the malicious code will not be decrypted and executed.

```
int __usercall patchParentProc @<eax>(DWORD a1 @<ecx>, int (*parentProcOffset)() @<esi>)
{
    int payloadRelativeoffset; // ebx
    DWORD oldProtect;          // [esp-10h] [ebp-14h]
    DWORD flOldProtect;        // [esp+0h] [ebp-4h]

    flOldProtect = a1;
    if (*parentProcOffset != 0x85 || *(parentProcOffset + 1) != 0xC0          // test eax, eax
        || *(parentProcOffset + 2) != 0xF || *(parentProcOffset + 3) != 0x84) // jz
    {
        return 0;
    }
    payloadRelativeoffset = decryptPayload - parentProcOffset - 5;
    VirtualProtect(parentProcOffset, PAGE_EXECUTE, PAGE_EXECUTE_READWRITE, &flOldProtect);
    *(parentProcOffset + 2) = BYTE1(payloadRelativeoffset);
    *(parentProcOffset + 3) = BYTE2(payloadRelativeoffset);
    oldProtect = flOldProtect;
    *(parentProcOffset + 1) = payloadRelativeoffset;
    *parentProcOffset = 0xE9; // jmp
    *(parentProcOffset + 4) = HIBYTE(payloadRelativeoffset);
    VirtualProtect(parentProcOffset, PAGE_EXECUTE, oldProtect, &flOldProtect);
    return 0;
}
```

Figure 1. Decompiled function responsible for patching the parent process

The difference between the original and patched hpqhvind.exe is shown in Figure 2.

```
CoInitialize(0);                                  | CoInitialize(0);
GetCommandLineW();                                | GetCommandLineW();
sub_401656(&v12);                                 | sub_371656(v5);
LOBYTE(v17) = 1;                                  | LOBYTE(v7) = 1;
hLibModule = LoadLibraryW(L"hpqhvsei.dll");       | hLibModule = LoadLibraryW(L"hpqhvsei.dll");
if (!hLibModule)                                  | JUMPOUT(0x66B01000); // patched by the malicous DLL
    goto LABEL_8;                                 |
```

Figure 2. Difference between original (left) and patched (right) hpqhvind.exe

The part of the code that is patched is located at the very beginning of the main function of hpqhvind.exe. As we can see in Figure 2, the patched code is located right after the load of hpqhvsei.dll. This means that the function responsible for decrypting and executing the payload is executed directly after the load of the malicious DLL.

Payload decryption

The encrypted payload is located in the .rdata section of hpqhvsei.dll and the decryption algorithm is an XOR loop where the XOR key is updated at each iteration, as shown in Figure 3.

```
xor_key = 0x409f6874
payload_size = 0x20a77
for i in range(payload_size):
    payload[i] = xor_key ^ payload[i]
    xor_key = 0x77*((xor_key << 0x10) + (xor_key >> 0x10)) + 0x13
```

Figure 3. Pseudocode of the payload decryption loop

The decrypted payload is the usual shellcode responsible for ShadowPad initialization (obfuscated using fake conditional jumps to hinder disassembly).

Persistence

After having been decrypted, ShadowPad's shellcode is executed. It will first achieve persistence on the system by writing the in-memory patched parent process to disk to a path specified in the configuration string pool. In the case we examined, the path was C:\ProgramData\DRM\CLR\CLR.exe. It then creates a service named clr_optimization_v4.0.30229_32, which is responsible for executing CLR.exe. To avoid suspicion, this service name, as well as the executable name, were chosen to look similar to the name of a Microsoft .NET optimization Service.

The full staging process is summarized in Figure 4. The numbering on each arrow corresponds to the chronological sequence of events.

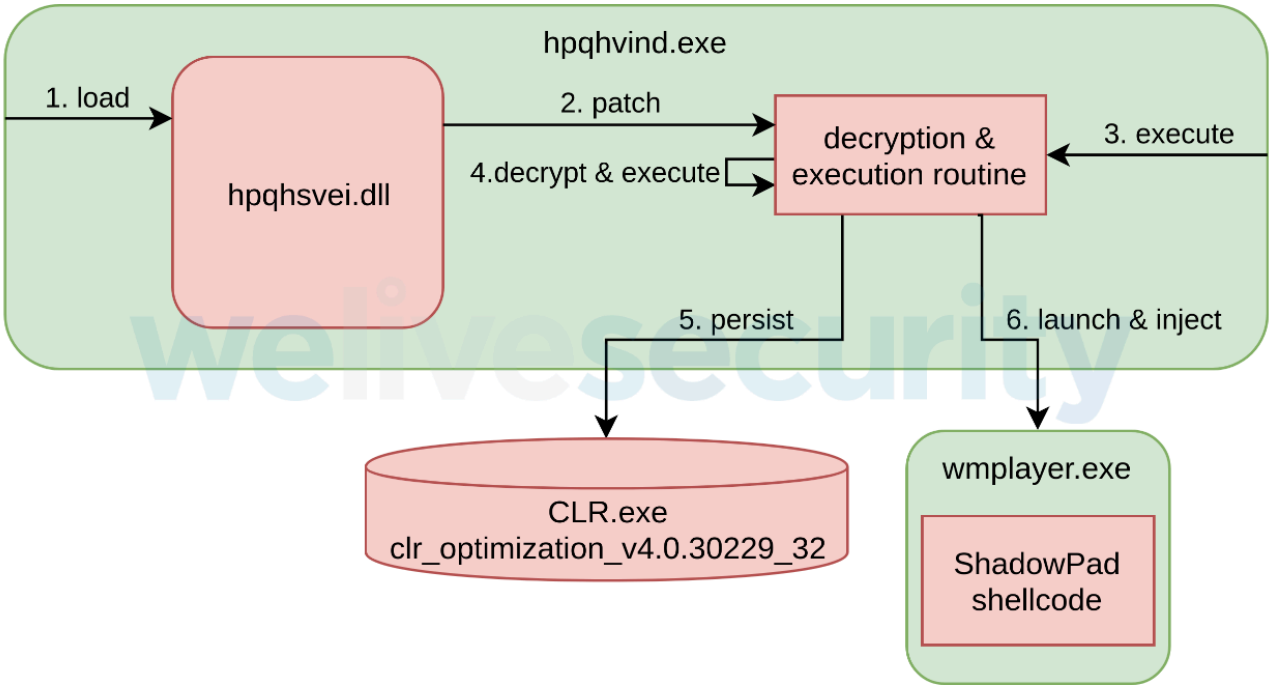

Figure 4. ShadowPad staging process

Modules

ShadowPad is a multimodular backdoor where the modules are referenced from the Root module with a circular list from which one can extract the module address, a UNIX timestamp (probably embedded automatically during the module's compilation process) and a module identifier. From the module itself we can also extract the name the developer gave to the module. This version embeds the 17 modules listed in the following table:

Table 1. Modules used with this ShadowPad version

ID	Name	Timestamp	Description
100	Root	Thu 24 Oct 2019 12:08:27 PM UTC	Initial shellcode
101	Plugins	Thu 24 Oct 2019 12:07:02 PM UTC	Provides API for the other modules; loads modules
102	Config	Thu 24 Oct 2019 12:07:09 PM UTC	Handles encrypted configuration string pool
103	Install	Thu 24 Oct 2019 12:07:46 PM UTC	Achieves persistence
104	Online	Thu 24 Oct 2019 12:07:17 PM UTC	Overall communications with the C&C server
106	ImpUser	Thu 24 Oct 2019 12:07:24 PM UTC	User impersonation via token duplication
200	TCP	Thu 24 Oct 2019 12:01:01 PM UTC	TCP communications
202	HTTPS	Thu 24 Oct 2019 12:01:15 PM UTC	HTTPS communications
207	Pipe	Thu 24 Oct 2019 12:01:35 PM UTC	Handles named pipes
300	Disk	Thu 24 Oct 2019 12:02:29 PM UTC	File system operations
301	Process	Thu 24 Oct 2019 12:02:36 PM UTC	Process handling
302	Servcie	Thu 24 Oct 2019 12:02:45 PM UTC	Service handling
303	Register	Thu 24 Oct 2019 12:02:52 PM UTC	Registry operations

ID	Name	Timestamp	Description
304	Shell	Thu 24 Oct 2019 12:03:00 PM UTC	Command line operations
306	Keylogger	Thu 24 Oct 2019 12:03:16 PM UTC	Keylogging to file system
307	Screen	Thu 24 Oct 2019 12:03:25 PM UTC	Screenshot capture
317	RecentFiles	Thu 24 Oct 2019 12:04:44 PM UTC	Lists recently accessed files

These modules, except for RecentFiles, have already been mentioned by Kaspersky and Avast. Notice the "Servcie" typo.

As usual, all the module timestamps are spread over a short time range, which could suggest the use of a build framework to compile these modules. This also suggests that these modules were built a few hours before the launcher itself, whose compilation timestamp is Thu Oct 24 14:10:32 2019. Since this compilation timestamp dates back two weeks before this campaign, it's likely that it hasn't been tampered with by the attackers.

One might also note that the number of modules embedded in this variant is much higher (17) than the number of modules embedded in the variants previously documented in our white paper (8 to 10 modules).

By default, every keystroke is recorded using the Keylogger module (306, previously documented by Avast) and saved to disk in the file %APPDATA%\PAGM\OEY\XWWEYG\WAOUE.

The log file is encrypted using the same algorithm as the one used to encrypt static strings from the module. Using this module by default indicates that the attackers are interested in stealing information from the victims' machines. In contrast, the variants we described in our white paper didn't even have that module embedded.

Configuration

As with previous ShadowPad variants, the Config module (102) contains an encrypted string pool that can be accessed from any other module. The string pool is never stored entirely decrypted in memory; the field of interest is decrypted when needed and then immediately freed (thus quickly unavailable). The configuration size is 2180 bytes and the encrypted strings are located at offset 0x84. The algorithm used to decrypt the strings is the same as the one used to decrypt the static strings of the module. The decrypted content of the string pool is the following:

0x84: 2019/11/7 16:28:36
0x99: **CAMPAIGN_ID_REDACTED**
0xa1: %ALLUSERSPROFILE%\DRM\CLR\CLR.exe
0xc5: clr_optimization_v4.0.30229_32
oxe6: clr_optimization_v4.0.30229_32
0x107: clr_optimization_v4.0.30229_32

0x128: SOFTWARE\Microsoft\Windows\CurrentVersion\Run
0x158: CLR
0x15e: %ProgramFiles%\Windows Media Player\wmplayer.exe
0x197: %windir%\system32\svchost.exe
0x1b7: TCP://b**[redacted]**.dnslookup.services:443
0x1db: UDP://b**[redacted]**.dnslookup.services:443
0x202: SOCKS4
0x21e: SOCKS5

The campaign ID is located at offset 0x99 and is the name of the targeted university. Having a campaign ID related to the target is quite common in the case of ShadowPad and Winnti.

Interestingly, the timestamp present in this config at offset 0x84 is later than the modules' timestamps and the loader compilation timestamp. This suggests that this config is added manually to the sample after having been built. Even though it's probably coincidental, the date within the config corresponds to the date of the first detection of this sample at the corresponding university.

Network Communications

Once installed on the system, ShadowPad starts a hidden and suspended Microsoft Windows Media Player wmplayer.exe process and injects itself into that process. The path to wmplayer.exe is provided by the Config module.

Once ShadowPad is injected into wmplayer.exe, the Online module will contact the C&C server using the URL specified in the configuration. It will then start listening for connections on port 13567 after having updated firewall rules accordingly:

Registry key:
HKLM\SYSTEM\ControlSet001\services\SharedAccess\Parameters\FirewallPolicy\FirewallRules\{816381AB-1400-45E5-B560-B8E11C5988CF}

Value:
v2.10|Action=Allow|Active=TRUE|Dir=In|Protocol=6|Profile=Public|LPort=13567|Name=Network Discovery (TCP)|

The communication is then handled by the TCP module (200), which was previously documented by Kaspersky.

Winnti malware was there as well

In addition to ShadowPad, the Winnti malware was found on some machines at these two universities at the end of October (i.e. two weeks before ShadowPad) in the file C:\Windows\System32\oci.dll and is detected by ESET products as Win64/Winnti.CA.

The Winnti malware usually contains a configuration specifying a campaign ID and a C&C URL. On all machines the campaign ID matches the name of the targeted university and the C&C URLs are:

- w**[redacted]**.livehost.live:443
- w**[redacted]**.dnslookup.services:443

where the redacted part corresponds to the name of the targeted university.

C&C URL format

One can observe that the C&C URL used by both Winnti and ShadowPad complies to the scheme [backdoor_type][target_name].domain.tld:443 where [backdoor_type] is a single letter which is either “w” in the case of the Winnti malware or “b” in the case of ShadowPad.

From this format, we were able to find several C&C URLs, including three additional Hong Kong universities’ names. The campaign identifiers found in the samples we’ve analyzed match the subdomain part of the C&C server, showing that these samples were really targeted against these universities.

Conclusion

The Winnti Group is still actively using one of its flagship backdoors, ShadowPad, this time against Hong Kong universities. In this campaign, the VMProtected launcher used with ShadowPad, as well as with the PortReuse backdoor and skip-2.0, was replaced by a simpler one. That these samples, in addition to having been found at these universities, contain campaign IDs matching the universities’ names and use C&C URLs containing the universities’ names are good indications that this campaign is highly targeted.

We will continue to monitor new activities of the Winnti Group and will publish relevant information on our blog. For any inquiries, contact us at threatintel@eset.com. The IoCs are also available in our GitHub repository.

Indicators of Compromise (IoCs)

ESET detection names

Win32/Shadowpad.C trojan
Win64/Winnti.CA trojan

File names

%ALLUSERSPROFILE%\DRM\CLR\hpqhvsei.dll
%ALLUSERSPROFILE%\DRM\CLR\CLR.exe
C:\windows\temp\hpqhvsei.dll
C:\windows\temp\hpqhvind.exe
%ALLUSERSPROFILE%\DRM\CLR\hpqhvsei.dll
%SYSTEM32%\oci.dll
%APPDATA%\PAGM\OEY\XWWEYG\WAOUE

Service display name

clr_optimization_v4.0.30229_32

C&C servers

b[org_name].dnslookup[.]services:443
w[org_name].livehost[.]live:443
w[org_name].dnslookup[.]services:443

ShadowPad launcher

Similar sample to avoid disclosing targeted universities.
693f0bd265e7a68b5b98f411ecf1cd3fed3c84af

MITRE ATT&CK techniques

Tactic	ID	Name	Description
Persistence	T1050	New Service	ShadowPad persists as a service called clr_optimization_v4.0.30229_32.
Defense Evasion	T1073	DLL Side-Loading	ShadowPad's launcher is loaded by a legitimate executable via DLL side-loading.
	T1055	Process Injection	ShadowPad is injected into a wmplayer.exe process.
	T1140	Deobfuscate/Decode Files or Information	ShadowPad launcher uses XOR to decrypt the payload. ShadowPad uses a custom algorithm to decrypt strings and configuration.
	T1027	Obfuscated Files or Information	ShadowPad shellcode is XOR-encoded and uses fake conditional jumps to hinder disassembly. ShadowPad's strings and configuration are encrypted. It also uses API hashing.
	T1143	Hidden Window	ShadowPad is injected into a wmplayer.exe process started in a hidden window.
Discovery	T1010	Application Window Discovery	ShadowPad's keylogging module lists application windows.
	T1083	File and Directory Discovery	ShadowPad's RecentFiles module lists files recently accessed.
Command and Control	T1071	Standard Application Layer Protocol	ShadowPad can use HTTP and HTTPS for C&C communications.
	T1043	Commonly Used Port	ShadowPad uses TCP:443 and UDP:443.

Tactic	ID	Name	Description
	T1065	Uncommonly Used Port	ShadowPad listens on port 13567.
	T1095	Standard Non-Application Layer Protocol	ShadowPad can use UDP and TCP for C&C communications.
	T1024	Custom Cryptographic Protocol	ShadowPad uses its own cryptographic protocol for C&C communications.
Collection	T1056	Input Capture	ShadowPad has a keylogging module.
	T1113	Screen Capture	ShadowPad has a screenshot module.
Exfiltration	T1022	Data Encrypted	Keystrokes recorded by the keylogging module are stored encrypted on disk.

Source: https://www.welivesecurity.com/2020/01/31/winnti-group-targeting-universities-hong-kong/