SILKLOADER

Journey of a Cobalt Strike beacon loader along the silk road

Researched and written by
Mohammad Kazem Hassan Nejad,
Bert Steppé, and Neeraj Singh
WithSecure™ Intelligence, March 2023

Contents

Introduction

Commercial and open-source command-and-control (C2) frameworks have become a staple in most adversary toolkits, with Cobalt Strike (CS) being one of the most popular. Such frameworks are often leveraged by threat actors to stage and conduct post-exploitation attacks in compromised client estates.

The prevalence of Cobalt Strike usage in attacks has precipitated a drive towards the creation of improved detection capabilities against it. Conversely, adversaries have responded to this by implementing their own detection evasion strategies. The most common of these include adding complexity to the auto-generated beacon or stager payloads via the utilization of packers, crypters, loaders, or similar techniques. While some threat actors rely on commercial crypters, others opt to develop their own custom crypters or take existing custom crypters into use.

During our investigations through several human-operated intrusions that resembled precursors to ransomware deployments, we came across an interesting Cobalt Strike beacon loader that leveraged DLL side-loading, which we're tracking as SILKLOADER. By taking a closer look at the loader, we found several activity clusters leveraging this loader within the Russian as well as Chinese cybercriminal ecosystems.

In this report we share technical analysis of SILKLOADER and highlight notable activity clusters where it was seen in our investigations.

Technical analysis

Overview

The initial SILKLOADER samples found were maliciously crafted libvlc.dll files designed to be dropped alongside a legitimate but renamed VLC binary. Execution of the binary causes the malicious DLL to be side-loaded. It is worth noting that side-loading malware through VLC Media Player is a technique that has previously been used by threat actors [1][2][3]. Operations leveraging DLL side-loading techniques to launch Cobalt Strike beacons such as LithiumLoader[4] have also been observed in the past.

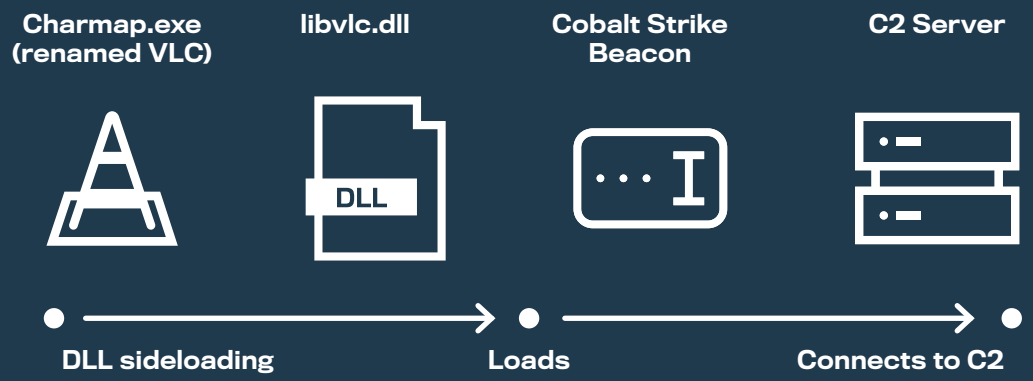


Figure 1: Renamed VLC loading malicious libvlc.dll.

1. https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/cicada-apt10-china-ngo-government-attacks
2. https://www.microsoft.com/en-us/security/blog/2018/11/08/attack-uses-malicious-inpage-document-and-outdated-vlc-media-player-to-give-attackers-backdoor-access-to-targets/
3. https://news.sophos.com/en-us/2022/11/03/family-tree-dll-sideloading-cases-may-be-related/
4. https://news.sophos.com/en-us/2022/11/03/family-tree-dll-sideloading-cases-may-be-related/

First stage (Shellcode loader)

Even though libvlc is an open-source project[5], the threat actor did not modify and compile the open-source code to create the malicious libvlc.dll files. Instead, the malicious DLLs were created from scratch. These DLLs are designed to mimic the legitimate libvlc.dll file by including export function names found in the legitimate version and thus imported by the VLC executable. Only one of the export functions contains malicious code. All other export functions are trivially implemented with "retn 0" instructions as shown in Figures 2 and 3 below.

The "libvlc_new" export function, which implements the malicious code, is called by the VLC executable to create a libvlc instance[6]. As such, the malicious code is executed immediately when the renamed VLC executable is launched. The malicious code acts as a shellcode loader.

The shellcode loader dynamically resolves the API functions it needs. These functions and module names are encrypted using a simple XOR based algorithm.

The loader first decodes the base64 encoded shellcode using the CryptStringToBinaryA function. It then performs a set of anti-analysis checks, which are explained in the following subsection. Lastly, it proxies the execution flow to the beginning of the shellcode by calling CertEnumSystemStoreLocation and setting the allocated shellcode's address as the callback function parameter. This is depicted in Figures 6 and 7.

5. https://code.videolan.org/videolan/vlc
6. https://videolan.videolan.me/vlc/group__libvlc__core.html#ga66f3850371cf5ece456f5c45313bb086

Base	Module	Address	Type	Ordinal	Symbol
00007FF727A60000	msdtc.exe	00007FFE103F17D8	Export	2	libvlc_new
00007FFE103F0000	libvlc.dll	00007FFE103F1868	Export	1	libvlc_add_intf
00007FFE200E0000	wininet.dll	00007FFE103F1868	Export	3	libvlc_playlist_play
00007FFE321A0000	dbgcore.dll	00007FFE103F1868	Export	4	libvlc_release
00007FFE32350000	dbghelp.dll	00007FFE103F1868	Export	5	libvlc_set_app_id
00007FFE34EA0000	apphelp.dll	00007FFE103F1868	Export	6	libvlc_set_user_agent
00007FFE356A0000	windows.storage.dll	00007FFE103F1868	Export	7	libvlc_wait

Figure 2: Renamed VLC loading malicious libvlc.dll.

```
; void libvlc_wait()
                public libvlc_wait
libvlc_wait     proc near               ; DATA XREF: __except_validate_context_record+7↓o
                                        ; .rdata:__guard_check_icall_fptr↓o ...
                retn    0               ; libvlc_add_intf
libvlc_wait     endp                    ; libvlc_playlist_play
                                        ; libvlc_release
                                        ; libvlc_set_app_id
                                        ; libvlc_set_user_agent
```

Figure 3: Export functions with retn 0

```
qmemcpy(&functionName, "uV@`ERJw[VYj>", 13);  // GetUserNameW
v12 = 0;
v13 = &functionName;
do
{
  *(_BYTE *)v13 ^= v12++ % 55u + 50;
  v13 = (__int128 *)((char *)v13 + 1);
}
while ( v12 < 13 );
*(_QWORD *)moduleName = '\v\v^FTBWs';          // Advapi32
moduleName[8] = 58;
v14 = 0;
v15 = moduleName;
do
  *v15++ ^= v14++ % 55u + 50;
while ( v14 < 9 );
GetUserNameW_dyn = resolve_api_function(moduleName, (LPCSTR)&functionName);
```

Figure 4: Example of API resolving

```
while ( v8 < 21 );
*moduleName = '9\n\x04BEMAq';                  // Crypt32
v11 = 0;
v12 = moduleName;
do
  *v12++ ^= v11++ % 55u + 50;
while ( v11 < 8 );
CryptStringToBinaryA_dyn = resolve_api_function(moduleName, &functionName);
if ( !(CryptStringToBinaryA_dyn)(b64EncodedShellC, 354456i64, 1i64, v6, &v33, 0i64, 0i64) )
  goto EXIT;
```

Figure 5: Base64 decoding

```
*functionName = xmmword_180017870;             // CertEnumSystemStoreLocation
v16 = '=)$)\t!1-';
v17 = 'M"$#';
v7 = 0;
v8 = functionName;
do
  *v8++ ^= v7++ % 0x37u + 50;
while ( v7 < 28 );
*moduleName = '9\n\x04BEMAq';                  // Crypt32
v9 = 0;
v10 = moduleName;
do
  *v10++ ^= v9++ % 0x37u + 50;
while ( v9 < 8 );
CertEnumSystemStoreLocation_dyn = resolve_api_function(moduleName, functionName);
(CertEnumSystemStoreLocation_dyn)(0i64, 0i64, *decodedShellC_addr);
```

Figure 6: Execution flow proxying via CertEnumSystemStoreLocation

Anti-sandbox checks

The loader contains three anti-sandbox checks that will cause execution to terminate:

1. It checks if the username (retrieved via GetUserNameW) is “vbccsb”. This is the default username used by ThreatBook Cloud Sandbox, a platform primarily used within the Chinese cybersecurity sphere.
2. It checks if the process command line contains the word “TRANSFER”. This is likely an anti-sandbox check for VirusTotal sandboxes.
3. It checks if the process name (VLC executable) matches a hard-coded value. This is likely an anti-sandbox check as well, as certain sandboxes do not use the original filename for execution. Based on analyzed samples, original filenames appear to be either one of following:

- msdtc.exe
- wpspdf.exe
- charmap.exe

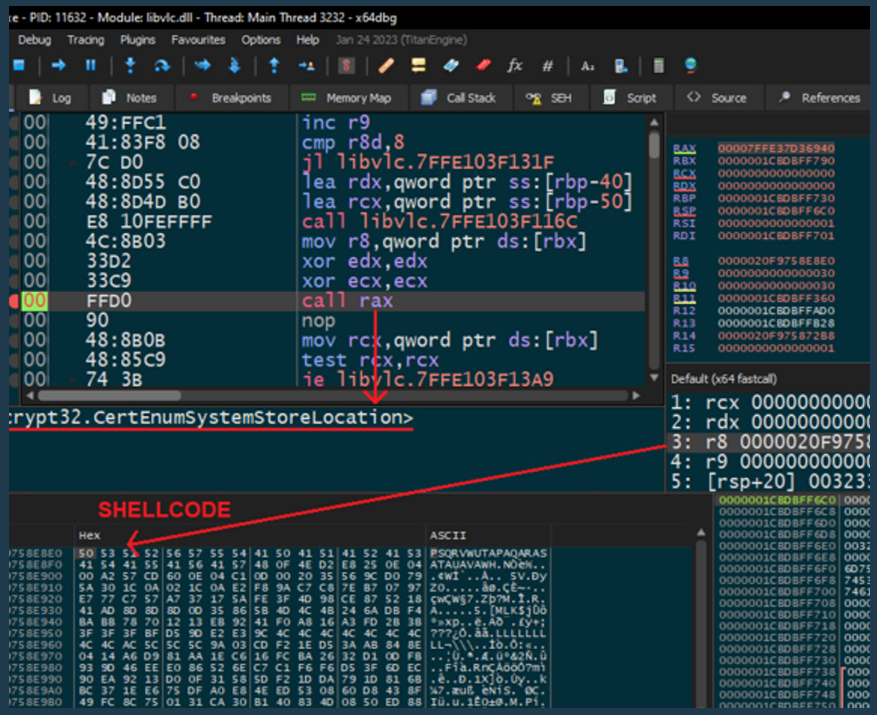


Figure 7: Execution flow transferred to shellcode

```
GetUserNameW_dyn = resolve_api_function(moduleName, &functionName);
if ( (GetUserNameW_dyn)(userName, &v34) )
{
  // check process filename matches msdtc.exe
  if ( compare(commandLineArgs, L"\\msdtc.exe") != -1 )
  {
    // copy to check later if username is vbccsb
    v19 = copy(vbccsb_UserName, L"vbccsb");
    *moduleName = 1;
    v20 = copy(&functionName, userName);
    v2 = 3;
    v21 = v19[2];
    if...
    v22 = *(v20 + 2);
    if...
    if...
    // check if TRANSFER string is found in command line args
    if ( compare(commandLineArgs, L"TRANSFER") == -1 )
    {
      v24 = 1;
LABEL_23:
      if...
      if...
      if...
      goto EXEC_SHELLC;
    }
  }
LABEL_22:
  v24 = 0;
  goto LABEL_23;
}
if ( compare(commandLineArgs, L"\\msdtc.exe") == -1 || compare(commandLineArgs, L"TRANSFER") != -1 )
{
```

Figure 8: Example of implemented anti-analysis checks

Second stage (Shellcode)

The loaded shellcode contains a stub which calls a decoder located at the end of the loaded shellcode to decode another stub.

This newly decoded stub then decodes an appended payload which has been XORed. This is depicted in Figure 9. In the case of all SILKLOADER samples that were analyzed, the appended payload was a Cobalt Strike reflective loader. This indicates that SILKLOADER was created for the sole purpose of being used as a Cobalt Strike beacon loader.

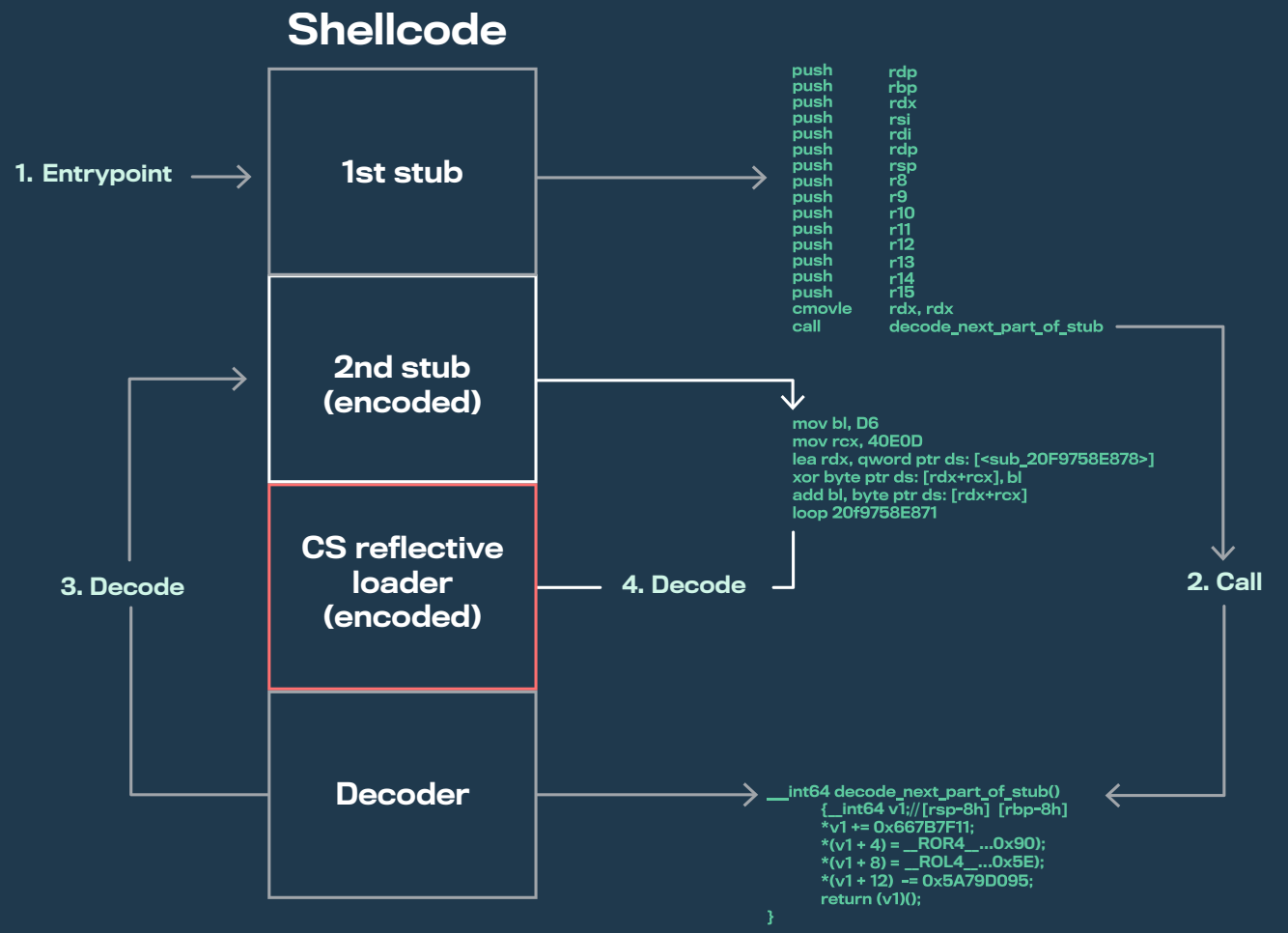


Figure 9: Cobalt Strike reflective loader unpacking.

Observed usage

The Discovery

WithSecure™ Intelligence analyzed several human-operated intrusions that occurred across various organizations throughout Asia, Europe, and Latin America in the last quarter of 2022. Although the attacks were contained before the threat actors could accomplish their objectives, the activities closely resembled precursors to ransomware deployments.

The intrusions across some of these organizations, namely in France, Brazil, and Taiwan, were similar to one another. The tooling and TTPs observed in these intrusions heavily overlapped with those reported to be used by operators deploying PLAY ransomware. These investigations led us to discover multiple Cobalt Strike beacon loaders, some of which we identified as SILKLOADER.

In some of the analyzed intrusions, SILKLOADER was used to gain a foothold in a client estate to conduct post-exploitation activities after initial access. For instance, WithSecure™ Intelligence analyzed an incident in a social welfare organization in France whereby the threat actor performed remote logon via a compromised Fortinet SSL VPN to stage multiple Cobalt Strike beacons in the client estate. The threat actor maintained a foothold in this organization for several months. During this time, they performed discovery and credential stealing activities, followed by deployment of multiple Cobalt Strike beacons. In one instance where the beacon was detected by WithSecure™ Elements Endpoint Protection (EPP), the threat actor staged another Cobalt Strike beacon that was packed by a different loader. However, after a few failed attempts due to EPP blocking, the attacker attempted to stage a malicious libvlc.dll file alongside a renamed VLC executable called msdtc.exe to launch their beacon. This was the first instance where we observed SILKLOADER.

In addition to SILKLOADER, the threat actor had dropped a Cobalt Strike beacon that was packed using a different loader, which was also specifically designed and used for Cobalt Strike beacons and beacon downloaders. We are tracking this separately as BAILLOADER. We confirmed that BAILLOADER was also used in a reported incident by Darktrace that led to Quantum ransomware[7] and spotted in a IcedID infection reported by Cybereason[8]. The relationship is depicted in Figure 10.

WithSecure™ Intelligence also verified that BAILLOADER resembles the crypter tracked by IBM Security as TRON crypter[9]. TRON crypter was reportedly developed and maintained by a group operating under ITG23, also known as CONTI Group, Wizard Spider, or Trickbot Group. Subgroups within CONTI Group are known to handle development and maintenance of multiple crypters which are used to encrypt malware and Cobalt Strike beacon loaders. UA-Cert[10] has also reported the usage of TRON crypters in cyber-attacks on Ukrainian state organizations since the start of the 2022 Ukraine conflict.

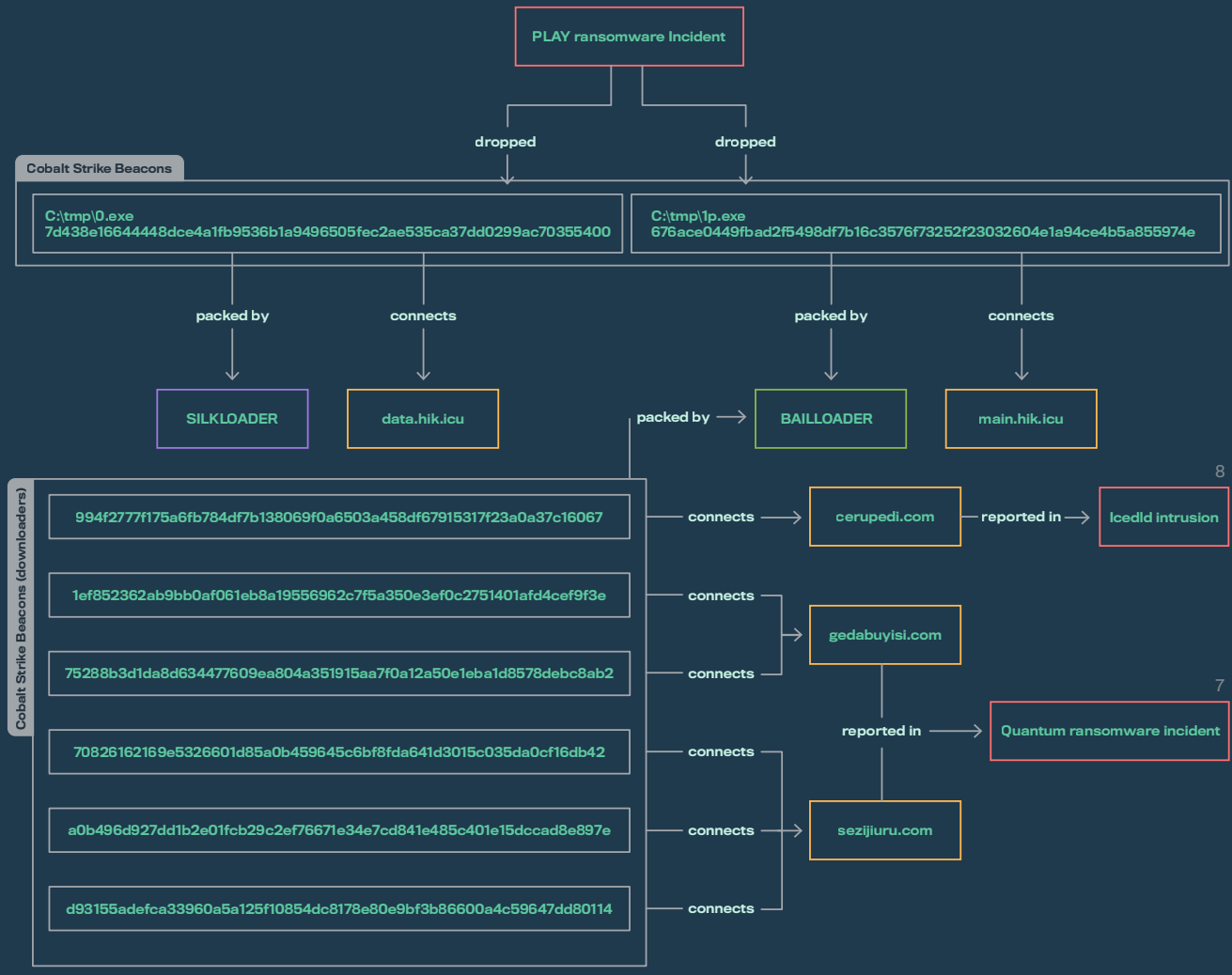


Figure 10: Connection between BAILLOADER and SILKLOADER

7. https://darktrace.com/blog/when-speedy-attacks-arent-enough-prolonging-quantum-ransomware
8. https://www.cybereason.com/blog/threat-analysis-from-icedid-to-domain-compromise
9. https://securityintelligence.com/posts/itg23-crypters-cooperation-between-cybercriminal-groups/
10. https://cert.gov.ua/article/39708

The Overlaps

The usage of SILKLOADER by PLAY ransomware operators was further confirmed by an incident response engagement reported by Sophos[11] where similar TTPs and the usage of SILKLOADER led to a confirmed deployment of PLAY ransomware.

Through retroactive hunting, WithSecure™ Intelligence found an additional SILKLOADER sample that shared the same parent domain with the Cobalt Strike team server configured in the SILKLOADER sample reported by Sophos. However, no additional evidence suggests that the SILKLOADER sample found through our retroactive hunting is associated to the PLAY ransomware group. Therefore, it is likely that different threat actors representing separate activity clusters share Cobalt Strike infrastructure and packer (SILKLOADER) provided by common affiliate(s). This relationship is depicted in Figure 11.

It is also worth noting that the Cobalt Strike infrastructure seen in the PLAY ransomware incident reported by Sophos differed significantly from other Cobalt Strike infrastructures we had observed with PLAY ransomware intrusions. This indicates the possibility of PLAY ransomware operators utilizing different Cobalt Strike beacons, crypters, and infrastructure provided by third-party affiliate(s). This was made further evident by the presence of BAILLOADER alongside SILKLOADER in a single intrusion.

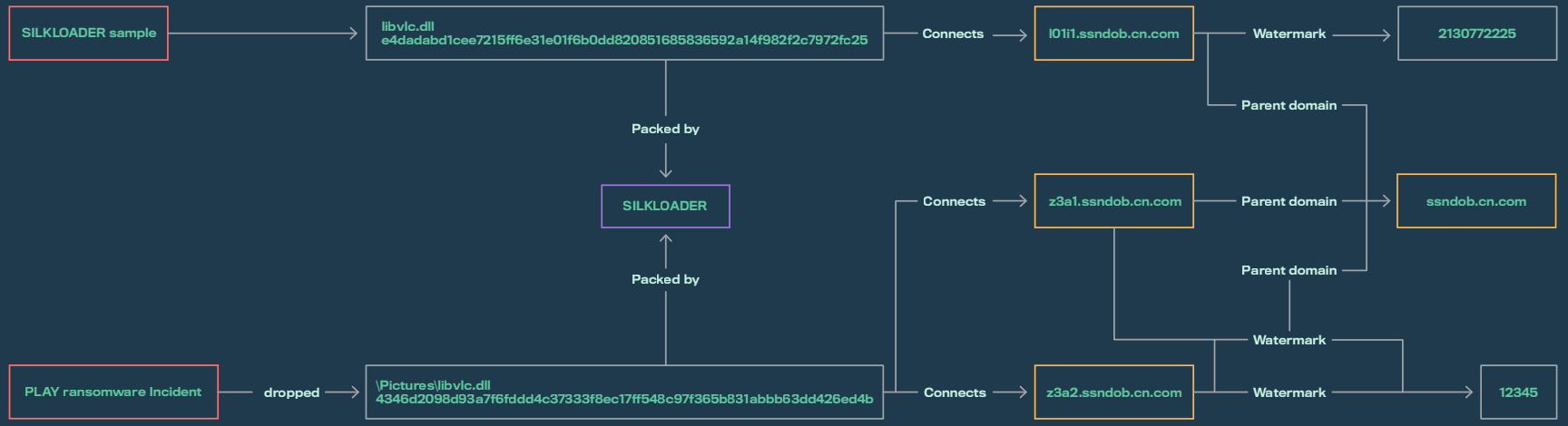


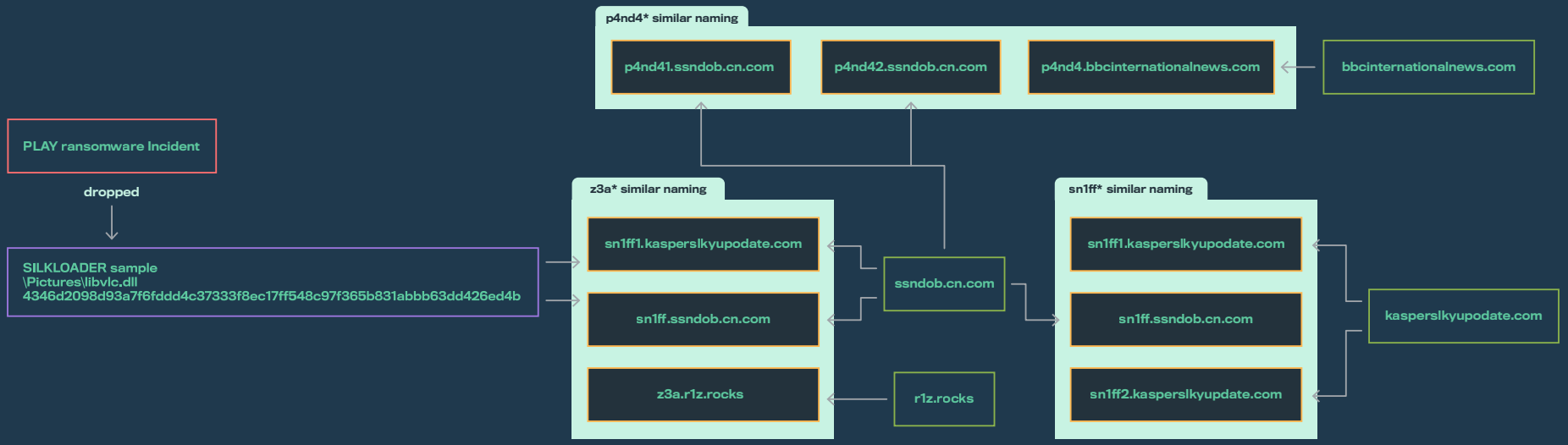
Figure 11. Connection between a PLAY ransomware incident and other unattributed SILKLOADER sample

11: https://infosec.exchange/@SophosXOps/109677906162017090

During retroactive hunting, WithSecure™ Intelligence found another SILKLOADER sample belonging to an unknown activity cluster that shared commonalities with the infrastructure seen in the SILKLOADER samples mentioned earlier in this article. Cobalt Strike beacons contain watermarks which identify them for licensing purposes. Ideally, all purchased licenses should be identifiable through unique watermarks. However, cracked, stolen, and modified versions of Cobalt Strike software can result in overlapping watermarks. This makes it difficult to track Cobalt Strike activities solely based on their watermarks. By using varying technical indicators, such as teamserver hashes, subdomains, domain names, and watermarks, we were able to find commonalities between different SILKLOADER samples. Figure 12 depicts a part of this.

Figure 12. Connection between unattributed SILKLOADER sample and a Cobalt Strike infrastructure

Our analysis of these SILKLOADER samples unraveled an expansive infrastructure that's setup to commonly host Cobalt Strike team servers which can be used amongst multipleactivity clusters and threat actors, for instance within PLAY ransomware intrusions or GootLoader[12] infections used for initial access into the systems. A small part of this Cobalt Strike infrastructure is depicted in Figure 13.



Ref: https://infosec.exchange/@SophosXOps/109677906162017090

Figure 13. Part of an expansive Cobalt Strike infrastructure

12. https://blog.nviso.eu/2022/07/20/analysis-of-a-trojanized-jquery-script-gootloader-unleashed
13. https://www.trendmicro.com/en_us/research/23/a/gootkit-loader-actively-targets-the-australian-healthcare-indust.html
14. https://www.cybereason.com/hubfs/THREAT%20ALERT%20GootLoader%20-%20Large%20payload%20leading%20to%20compromise%20(BLOG).pdf

In January 2023, TrendMicro reported that an instance of libvlc.dll, which we verified as SILKLOADER, was delivered through GootLoader[13]. Although the final payload was unknown in this case, we were able to determine that the same infrastructure was used across the Cobalt Strike team server seen in this instance of SILKLOADER and those used in other suspected PLAY ransomware intrusions where SILKLOADER was used. Since there are no known instances of GootLoader being used for initial access to deploy PLAY ransomware, it is likely that these represent separate activity clusters, and the threat actors share Cobalt Strike infrastructure and that SILKLOADER is likely provided by common affiliate(s). This relationship is depicted in Figure 14. In February 2023, Cyberreason also reported an incident related to GootLoader[14] which used libvlc.dll via sideloading. We have verified this sample to be SILKLOADER as well.

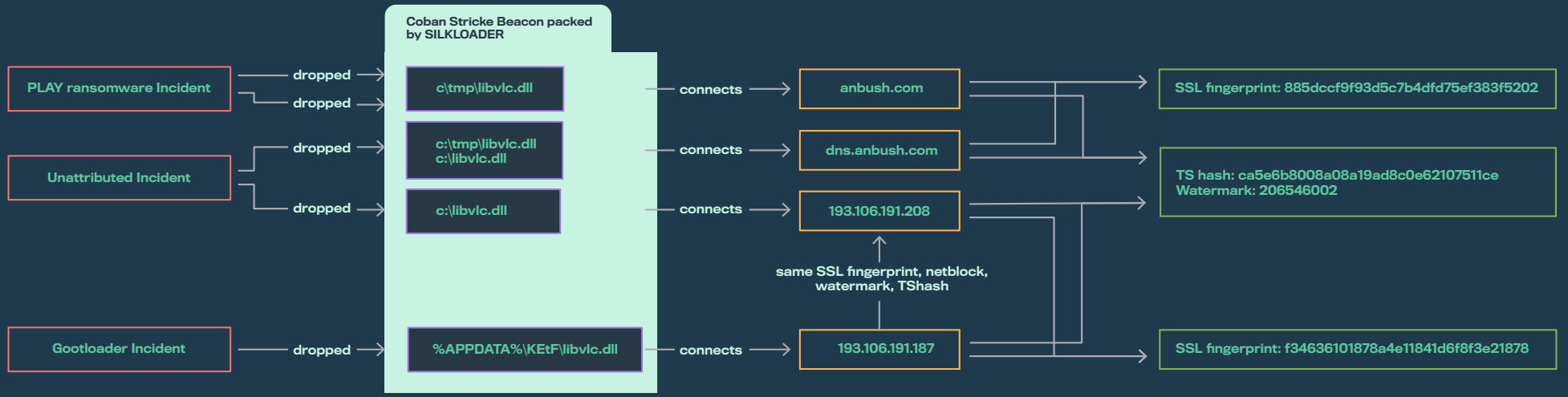


Figure 14. Common infrastructure seen between SILKLOADER samples in a PLAY ransomware intrusion and GootLoader infection

In the past, one could consider an intrusion-set to almost conform to a set of threat actors operating directly as part of an organizational structure. However, with the introduction of ransomware-as-a-service (RaaS), this is no longer the case. Recent changes to the cybercrime landscape have driven a more fractured and service-oriented approach to ransomware operators. This has made it increasingly difficult to correlate tools and TTPs to specific ransomware variants. The initial access broker marketplace is now thriving in the underground economy, whereby accesses to networks deemed lucrative are being increasingly proliferated to actors affiliated with multiple ransomware variants.

We see examples of that ecosystem in action in the above figures, which illustrate Cobalt Strike infrastructure we believe to be inter-related. This infrastructure is being used to service multiple intrusions utilizing different TTPs and different ransomware variants. The same can be said for tooling, as more defined ransomware groupings disintegrate and multiple successors emerge, tool usage will cross-pollinate through different variants as they are either retained by individual actors, released, or leaked.

Based on our findings, WithSecure™ Intelligence assesses with medium confidence that a malicious actor from the Russian cybercriminal ecosystem procured SILKLOADER either as a builder or direct source code. We believe that SILKLOADER is currently being offered as an off-the-shelf loader through a Packer-as-a-Service program to Russian-based threat actors. This loader is being provided either directly to ransomware groups or possibly via groups offering Cobalt Strike/Infrastructure-as-a-Service to trusted affiliates. Most of these affiliates appear to have been part of or have had close working relationships with the CONTI group, its members, and offspring after its alleged shutdown. It is plausible that the threat actor has created a build machine to automatically crypt beacons with SILKLOADER. This idea was validated by ContiLeaks, where it was suggested that the CONTI Group[15] employed these sorts of infrastructures in the past for their own use.

Analysis described so far in this report has focused on recent SILKLOADER activity. However, we were interested in understanding the malware's origin and thus performed retroactive analysis on older samples. In the next section of this document, we'll describe our findings related to SILKLOADER's origins.

Contained Resources By Language

CHINESE SIMPLIFIED	10
ENGLISH US	1

Contained Resources

SHA-256	File Type	Type	Language	Entropy	Chi2
5080cdb4681299a1d73721a88147a8de89315a9bc80e235c2689326f8c50a85e	DOS EXE	GUP	CHINESE SIMPLIFIED	6.34	11022403
262d966fd82312bcb82b056b2dc378b173f5c335917bc53125aef7f5a03cfce4	DOS EXE	LIB	CHINESE SIMPLIFIED	6.55	1357606.12
92717951aae89e960b142cef3d273f104051896a3d527a78ca4a88c22b5216a5	XML	XML	CHINESE SIMPLIFIED	5.01	53282.98
9e4372979b69241ac2fbb56857b18b1e23b4b14b30b11142d955e0ed839dfeb1	unknown	RT_ICON	CHINESE SIMPLIFIED	4.33	109080.7
c4810ca3c47864ee1afe2945c1ceb8d8bfa089076c1482af684dee2f4fa9e262	unknown	RT_ICON	CHINESE SIMPLIFIED	5.38	37951.05
f0094827dd717591eefeeb08722538ca2a9e86191293a8e448775d65c48bbf50	unknown	RT_ICON	CHINESE SIMPLIFIED	5.94	68302.63

Figure 15: Resources seen in a dropper containing GUP.exe and libcurl.dll

15. https://securityintelligence.com/posts/itg23-crypters-cooperation-between-cybercriminal-groups/

The Origin

WithSecure™ Intelligence identified earlier instances of SILKLOADER dating back to early 2022. The loader was spotted in multiple distinct activity clusters linked to threat actors and victims within the East Asian region, predominantly in Hong Kong and China.

Notably, instead of libvlc.dll, the earliest instance of identical loader code was found in a malicious libcurl.dll file that was sideloaded by GUP.exe, which is part of Notepad++ application. “libcurl.dll” and “GUP.exe” were present in the resource section of files named 安全查杀工具.exe (security scanning tool.exe) and 违规投诉–证据.pdf.exe (Violation Complaint – Evidence.pdf.exe).

Upon execution of these files, GUP.exe, libcurl.dll, and gup.xml are dropped into %temp% folder and GUP.exe is automatically executed, which sideloads libcurl.dll. Based on our analysis of the libcurl.dll sample, the “curl_easy_init” export function contained malicious code identical to the libvlc.dll variant of SILKLOADER mentioned earlier in this report. The configured Cobalt Strike team server was pic1.update3.mypicture[.]info.

Unlike other instances of SILKLOADER, where the loader had likely been created in isolation and distributed to different threat actors, in this case, both the loader and dropper were created by the same threat actor. This is made evident by the usage of the same linker and compiler used for both the dropper (安全查杀工具.exe) and loader (libcurl.dll) as well as the compilation timestamp of both files, which differed by a few minutes.

The next spotted instance of SILKLOADER was attributed to the same activity cluster and it was the first time libvlc.dll variant of SILKLOADER was observed. The VLC executable was renamed as charmap.exe and the configured Cobalt Strike team server was update3.mypicture[.]info.

Based on the information above, we assess with medium confidence that the adversary representing this activity cluster is the creator of SILKLOADER. This evidence is also backed by the fact that this is the first known activity cluster where SILKLOADER was seen. Additionally, this is the only known activity cluster where SILKLOADER was compiled as libcurl.dll.

The next activity cluster where SILKLOADER was observed in, relied on renamed VLC executable file names charmap.exe and wpspdf.exe respectively. In both instances, the configured Cobalt Strike team server was cdn.goby[.]pw. Based on network indicators, the same threat actor had been observed exploiting VMWare vulnerabilities in the wild[16].

Furthermore, WithSecure™ Intelligence was able to identify numerous Cobalt Strike beacons written in Go as well as Chinese-based tooling such as NPS[17] and custom SOCKS proxy tool seen in the Chinese cybercriminal ecosystem being used by the threat actor.

The last activity cluster where SILKLOADER was observed within the Chinese cybercriminal ecosystem, utilized a malicious spam campaign. This spam campaign predominantly targeted companies in China[18]. The following are some examples of attachment names used in this campaign:

1. 民航总局员工信息_结构组成(内部绝密).rar - CAAC employee information_structure composition (internal top secret).rar
2. 微信客户端0day详情通告.zip - Wechat Client 0day Details Announcement.zip
3. 薪酬调整计划通知.rar - Salary Adjustment Plan Notice.rar
4. jddj电视节目宣传需求.rar - jddj TV program publicity needs.rar

SILKLOADER was found inside one of the attachments “民航总局员工信息_结构组成(内部绝密).rar” in this cluster. The dropper used in this campaign was delivered as an attachment in a malicious spam email. Files libvlc.dll and renamed vlc.exe (charmap.exe) were stored inside the cursor directory of the resource of the dropper present inside the email attachment, as depicted in Figure 16.

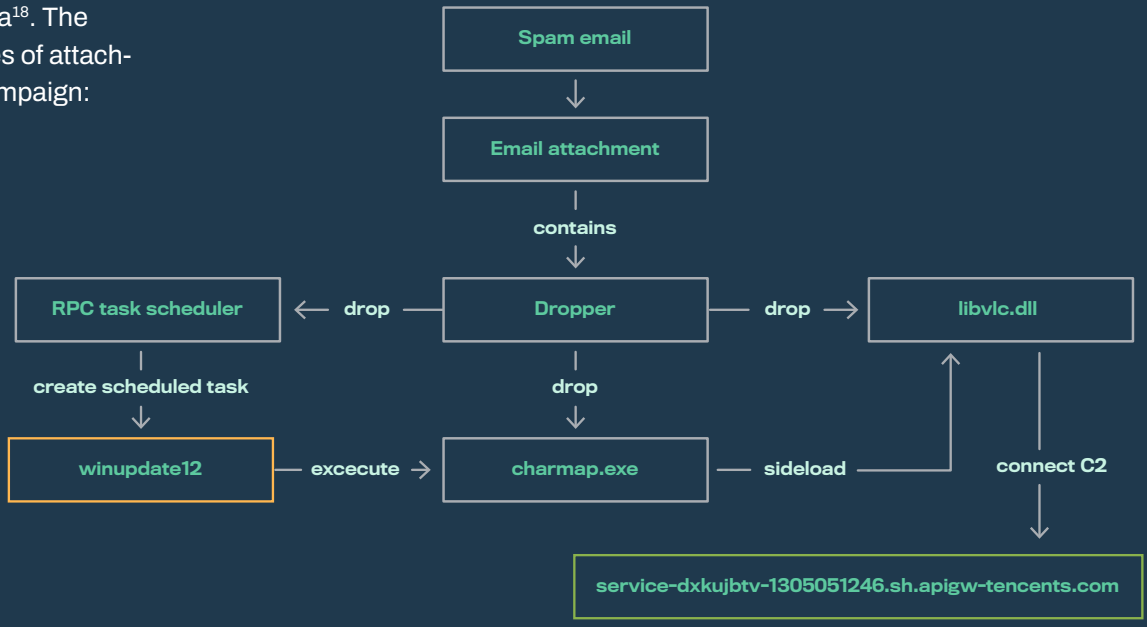


Figure 16. Infection chain

16. https://unit42.paloaltonetworks.com/atoms/vmware-vulnerabilities/
17. https://github.com/ehang-io/nps
18. https://twitter.com/h2jazi/status/1554890487806451714

The dropper does not directly execute the VLC executable to side-load SILKLOADER. Instead, it uses a custom binary that is dropped which acts as an RPC-based task scheduler, which creates a scheduled task to subsequently run the renamed VLC executable, charmap.exe, which sideloads libvlc.dll. The RPC task scheduler accepts a path to a file, a task name, and a number of hours after which the scheduled task should run as parameters. An example of this might be “C:\Users\Public\Libraries\tsc.exe %APPDATA%\charmap.exe winupdate12 1”. The configured Cobalt Strike team server was service-dxku-jbtv-1305051246.sh.apigw.tencentcs[.]com.

In other observed instances, the RPC task scheduler was used to add scheduled tasks for VBS and LNK files as well as other unidentified malware. Some notable C2s observed in these instances were hc32.weixinzx[.]org , sc32.hserver.dns-dns[.]com, hc64.hserverdns[.]com, hs.hserverdns[.]com.

At the time of writing, indicators found across these activity clusters were found to be unique and unattributable to known threat actors. Other than the usage of SILKLOADER, no other overlapping commonalities could be observed between them. However, the geographical targeting, compilation timestamps, filenames, and other technical indicators were all consistent with Chinese-based operations.

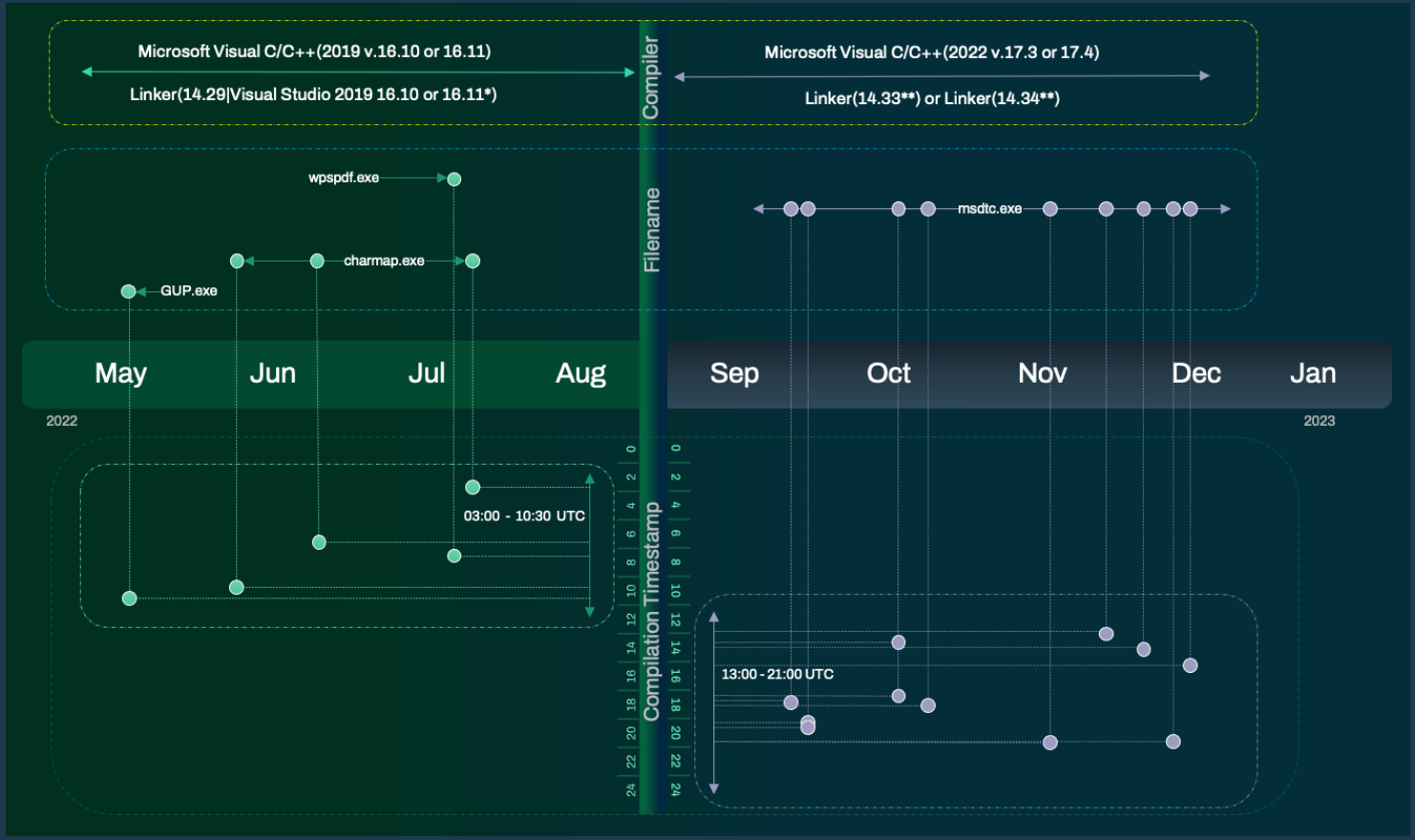


Figure 17. Timeline of SILKLOADER samples analyzed by WithSecure™

The Journey

We observed considerable changes in the usage and compilation of SILKLOADER samples after July 2022. All SILKLOADER samples found after this time relied on the VLC executable to be renamed as msdtc.exe as opposed to previously used filenames such as wpspdf.exe and charmap.exe.

Compilation timestamps in analyzed samples did not appear to have been tampered with and the timestamps found in earlier cases were between 0300 and 1100 UTC. Those found later were between 1200 and 2100 UTC. A timeline of SILKOADER sample file names and compilation timestamps is depicted in Figure 17.

The rich header information found in SILKLOADER samples suggest that the binaries were compiled with different MSVC compiler and linker versions after the July 2022 break. Samples spotted before July 2022 were predominantly compiled using Microsoft Visual C/C++(2019 v.16.11) and some with Microsoft Linker(14.16|Visual Studio 2017 15.9*), whereas samples found after July 2022 were compiled with Microsoft Visual C/C++ (2022 v.17.3 and v.17.4).

We posit with medium confidence that SILKLOADER was originally written by threat actors acting within the Chinese cybercriminal ecosystem. The loader was used by the threat actors within this nexus atleast as early as May 2022 till July 2022. The builder or source code was later acquired by a threat actor within the Russian cybercriminal ecosystem between July 2022 and September 2022. This assessment is based on the following:

- The loader includes an anti-analysis method designed to detect and evade ThreatBook Cloud Sandbox, which is a solution predominantly used within the East Asian region, indicating that the loader was originally intended for this particular region.
- File names and spam e-mails associated with the earliest activity clusters appeared to target East Asian victims, predominantly in Hong Kong and China. It is not entirely clear how each group represented in the activity clusters within the Chinese cybercriminal ecosystem acquired this loader. However, all SILKLOADER-related incidents that were identified after July 2022 were no longer attributable to any activity cluster within the Chinese cybercriminal ecosystem, but rather Russian one.
- SILKLOADER compilation timestamps prior to July 2022 consistently occurred within afternoon-to-evening time zones in China. While post-September 2022 compilation timestamps occurred during evening time zones in Western Russia.
- VLC media player executable, which is used to side-load libvlc.dll was renamed to wpspdf.exe or charmap.exe in earlier instances. While post-September 2022, the file was consistently named msdtc.exe.
- No samples were found during the two-month gap between July 2022 and September 2022. This is likely the time that the loader source code or builder was sold or handed over. We speculate that the original Chinese author sold the loader to a Russian threat actor once they no longer had any use for it.

Conclusion

WithSecure™'s assessment is that SILKLOADER was originally created within the Chinese cybercriminal ecosystem. It was either sold or given to a threat actor within the Russian cybercriminal ecosystem between July and September 2022. This indicates that cybercrime, and especially financially motivated cybercrime, is not geographically bound. In this case, it is more likely that the author was an independent coder who sold their tool on an underground forum[19]. Such components can and are sold or handed over to other groups when the situation favors such a transaction.

Of note, during our investigation, we discovered additional related samples. Unfortunately, we didn't have access to them. As new samples emerge, hypotheses, proposed timelines, and activity clusters can all change. New information may even precipitate the emergence of new activity clusters. As such, our current conclusions may only be valid at the time of writing.

We believe that SILKLOADER is currently being offered as an off-the-shelf loader through a Packer-as-a-Service program to Russian-based threat actors. This loader is being provided either directly to ransomware groups or possibly via groups offering Cobalt Strike/Infrastructure-as-a-Service to trusted affiliates. Most of these affiliates appear to have been part of or have had close working relationships with the CONTI group, its members, and offspring after its alleged shutdown.

As the cybercriminal ecosystem becomes more and more modularized via service offerings[20], it is no longer possible to attribute attacks to threat groups simply by linking them to specific components within their attacks. As demonstrated in this report, SILKLOADER was used by multiple groups, who are performing human-operated intrusions and deploying ransomware such as PLAY or using initial access provider malware such as GootLoader.

SILKLOADER started off as something small and obscure enough to fly under the radar. But it's now become part of a much larger and more impactful ecosystem. As Infrastructure-as-a-Service operations gain popularity, we'll continue to see similar looking custom packers and loaders employed by multiple adversarial groups.

19. https://intel471.com/blog/china-cybercrime-undergrond-deepmix-tea-horse-road-great-firewall
20. https://www.microsoft.com/en-us/security/blog/2022/05/09/ransomware-as-a-service-understanding-the-cybercrime-gig-economy-and-how-to-protect-yourself/

Acknowledgement

The authors wish to thank Timothy West and Andrew Patel for additional insights and content for this report.

Recommendations and protection

WithSecure™ Elements[21] Endpoint Detection and Response as well as WithSecure™ Countercept Detection and Response[22] detects multiple stages of the attack lifecycle. These will generate a single incident with detailed detections.

WithSecure™ Elements Endpoint protection offers multiple detections that detect the malware and its behavior. Ensure that real-time protection as well as DeepGuard are enabled. You may run a full scan on your endpoint. Our products currently offer the following detections against the malware:

- Trojan:W32/Hijacker.
- Trojan:W32/MalPayload.
- Trojan:W32/BailLoader.

If you believe your business has been targeted or fallen victim to the same or similar attack and require assistance, you can reach out to our 24/7 incident hotline[23].

21. https://www.withsecure.com/en/solutions/software-and-services/elements#trial
22. https://www.withsecure.com/en/solutions/managed-services/countercept
23. https://www.withsecure.com/en/about-us/company-contacts/24-7-incident-hotline

Indicators of Compromise (IOCs)

You can find an updated list of IOCs as well as YARA rules in WithSecure™ Lab's GitHub[24]

Samples - All Activity Clusters

SHA256	0248572780e94f3557c50d2c161365f09b175438ed5e750ccc5b5f1c895118c2	SILKLOADER libvlc.dll
	2e57d2d14d8f98464e501d99dad0ae2c2f237b45aceecb73850c17ad1455e39c	SILKLOADER libvlc.dll
	3d1df6633ec9b16f856c6ddf7c40138e524f7c5e286af2271e32643f88f071a2	SILKLOADER libvlc.dll
	4346d2098d93a7f6fddd4c37333f8ec17ff548c97f365b831abbb63dd426ed4b	SILKLOADER libvlc.dll
	54d45b872ee6651d670c3580da74ca56f626bc6ed5ce60cc7e3ba71bb68cd23f	SILKLOADER libvlc.dll
	5623f7b2a3e459d91ed85d20c20b58fb7edc166c63a3a72b703d7af400bdc12c	SILKLOADER libvlc.dll
	56377607abfb0616fc1dc222cc765966954155b69b5c4b16cad92cdd353720a4	SILKLOADER libvlc.dll
	575caa641dbb83364e8c0664737666f3bcd24d40316ff149c9ef476115b927cb	SILKLOADER libvlc.dll
	7070b232de836ce33b7778f6b6f7aecb252d542831367bc78d0eea77461be9a7	SILKLOADER libvlc.dll
	7c2ea97f8fff301a03f36fb6b87d08dc81e948440c87c2805b9e4622eb4e1991	SILKLOADER libvlc.dll
	7d438e1664448dce4a1fb9536b1a9496505fec2ae535ca37dd0299ac70355400	SILKLOADER libvlc.dll
	972ab4694d8177e65de6aef5b6eb0c1e1cafd1cad7bdea484e37ffb156184f34	SILKLOADER libvlc.dll
	a6fff6890c0b6642931f9b0bfd67f830bb85af0f218280776170abfcc5baa576	SILKLOADER libvlc.dll
	c83ac6dc96febd49c7c558e8cf85dd8bcb3a84fdc78b3ba72ebf681566dc1865	SILKLOADER libvlc.dll
	d77a59e6ba3a8f3c000a8a8955af77d2898f220f7bf3c0968bf0d7c8ac25a5ad	SILKLOADER libvlc.dll
	e4dadabd1cee7215ff6e31e01f6b0dd820851685836592a14f982f2c7972fc25	SILKLOADER libvlc.dll
	f1f5142a456e8a316b281ad7f2fe1b463d93f30460a6be3afa9c5593f1392656	SILKLOADER libvlc.dll
	262d966fd82312bcb82b056b2dc378b173f5c335917bc53125aef7f5a03cfce4	SILKLOADER libvlc.dll

24. https://github.com/WithSecureLabs/iocs/tree/master/SILKLOADER/

Samples - All Activity Clusters

SHA256	1ef852362ab9bb0af061eb8a19556962c7f5a350e3ef0c2751401afd4cef9f3e	BAILLOADER
	676ace0449fbad2f5498df7b16c3576f73252f23032604e1a94ce4b5a855974e	BAILLOADER
	70826162169e5326601d85a0b459645c6bf8fda641d3d3015c035da0cf16db42	BAILLOADER
	75288b3d1da8d634477609ea804a351915aa7f0a12a50e1eba1d8578debc8ab2	BAILLOADER
	994f2777f175a6fb784df7b138069f0a6503a458df67915317f23a0a37c16067	BAILLOADER
	a0b496d927dd1b2e01fcb29c2ef76671e34e7cd841e485c401e15dccad8e897e	BAILLOADER
	d93155adefca33960a5a125f10854dc8178e80e9bf3b86600a4c59647dd80114	BAILLOADER
	0322f9201a3887659f7568f3f2292248e29afc19a6e80cdda6915834a3fc925d	RPC Task Scheduler
	9c1914143b0ba7fa15848223d0695664fc8225c37eed09eeb00e0af1b7ee0d7b	SILKLOADER dropper
	bf02668f884937f697af326a6678fe5c1d3844e104da8c4049f23d63dcb5bb5c	SILKLOADER dropper
	d378d01f863454911a345869b66d60769a894c74dfebaa0a9a07efc884a3d15c	SILKLOADER dropper

Network - All Activity Clusters

IP	198.98.53.34
	193.106.191.187
	193.106.191.208
Domain	020-rce500.r1z.rocks
	cerupedi.com
	d3-up.ssndob.cn.com
	data.hik.icu
	dl.kaspersky360.com
	dl.kasperskyupdates.com
	dns.anbush.com
	gedabuyisi.com
	hc32.weixinzx.org
	hc64.hserverdns.com
	hs.hserverdns.com

Network - All Activity Clusters

Domain	l01i1.ssndob.cn.com
	main.hik.icu
	p4nd4.bbcinternationalnews.com
	p4nd41.ssndob.cn.com
	p4nd42.ssndob.cn.com
	sc.hserverdns.com
	sc32.hserver.dns-dns.com
	service-d9pbyhs4-1305051246.gz.apigw.tencentcs.com
	service-dxkujbtv-1305051246.sh.apigw.tencentcs.com
	sezijiru.com
	sn1ff.ssndob.cn.com
	sn1ff1.kasperslkyupdate.com
	sn1ff2.kasperslkyupdate.com
	upd.kasperskyupdates.com
	update-2.kaspersky360.com
	z3a.r1z.rocks
	z3a1.ssndob.cn.com
	z3a1.ssndob.cn.com

Who We Are

WithSecure™ is cyber security's reliable partner. IT service providers, MSSPs and businesses along with the largest financial institutions, manufacturers, and thousands of the world's most advanced communications and technology providers trust us for outcome-based cyber security that protects and enables their operations. Our AI-driven protection secures endpoints and cloud collaboration, and our intelligent detection & response is powered by experts who identify business risks by proactively hunting for threats and confronting live attacks. Our consultants partner with enterprises and tech challengers to build resilience through evidence-based security advice. With more than 30 years of experience in building technology that meets business objectives, we've built our portfolio to grow with our partners through flexible commercial models.

WithSecure™ is part of F-Secure Corporation, founded in 1988, and listed on the NASDAQ OMX Helsinki Ltd.