Hack the Real Box: APT41's New Subgroup Earth Longzhi

By By: Hara Hiroaki, Ted Lee Nov 09, 2022 Read time: 10 min (2749 words)

Published: 2022-11-09 · Archived: 2026-04-05 12:38:52 UTC

APT & Targeted Attacks

We looked into the campaigns deployed by a new subgroup of advanced persistent threat (APT) group APT41, Earth Longzhi. This entry breaks down the technical details of the campaigns in full as presented at HITCON PEACE 2022 in August.

In early 2022, we investigated an incident that compromised a company in Taiwan. The malware used in the incident was a simple but custom Cobalt Strike loader. After further investigation, however, we found incidents targeting multiple regions using a similar Cobalt Strike loader. While analyzing code similarities and tactics, techniques, and procedures (TTPs), we discovered that the actor behind this attack has been active since 2020. After clustering each intrusion, we concluded that the threat actor is a new subgroup of advanced persistent threat (APT) group APT41news- cybercrime-and-digital-threats that we call Earth Longzhi. In this entry, we reveal two campaigns by Earth Longzhi from 2020 to 2022 and introduce some of the group's arsenal in these campaigns. This entry was also presented at the HITCON PEACE 2022 conference in August this year.

Campaign overview

Since it first started being active in 2020, Earth Longzhi's long-running campaign can be divided into two based on the range of time and toolset. During its first campaign deployed from 2020 to 2021, Earth Longzhi targeted the government, infrastructure, and health industries in Taiwan and the banking sector in China. In its second campaign from 2021 to 2022, the group targeted high-profile victims in the defense, aviation, insurance, and urban development industries in Taiwan, China, Thailand, Malaysia, Indonesia, Pakistan, and Ukraine.

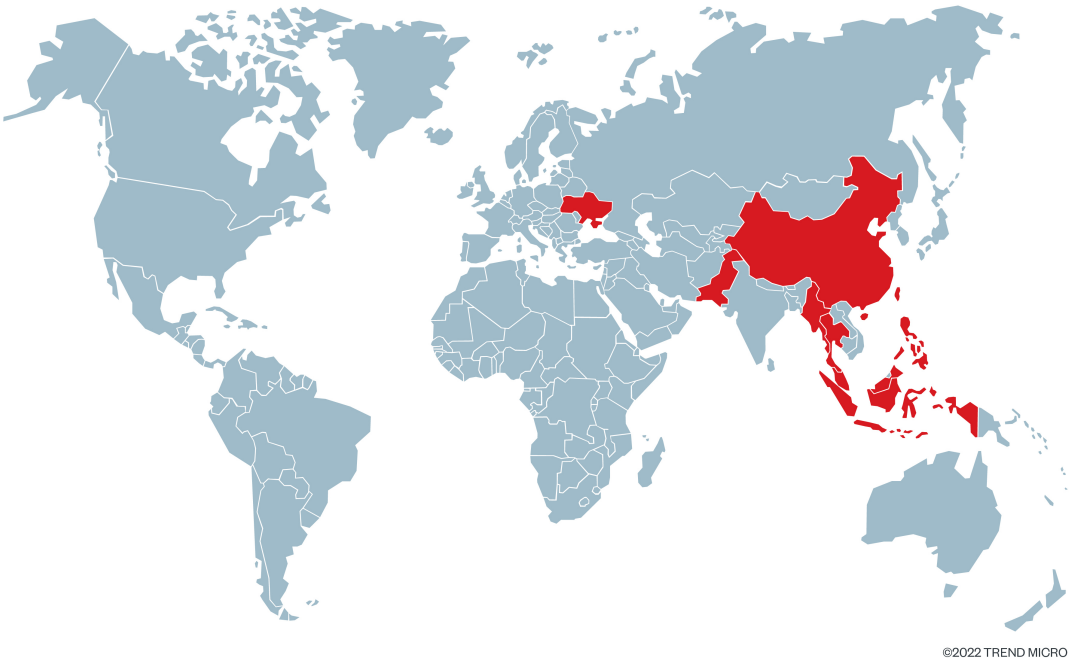

Figure 1. Earth Longzhi's victim countries from 2020 to 2022

Attack vector

Both campaigns used spear-phishing emails as the primary entry vector to deliver Earth Longhzhi's malware. The attacker embeds the malware in a password-protected archive or shares a link to download a malware, luring the victim with information about a person. Upon opening the link, the victim is redirected to a Google Drive hosting a password-protected archive with a Cobalt Strike loader we call CroxLoader.

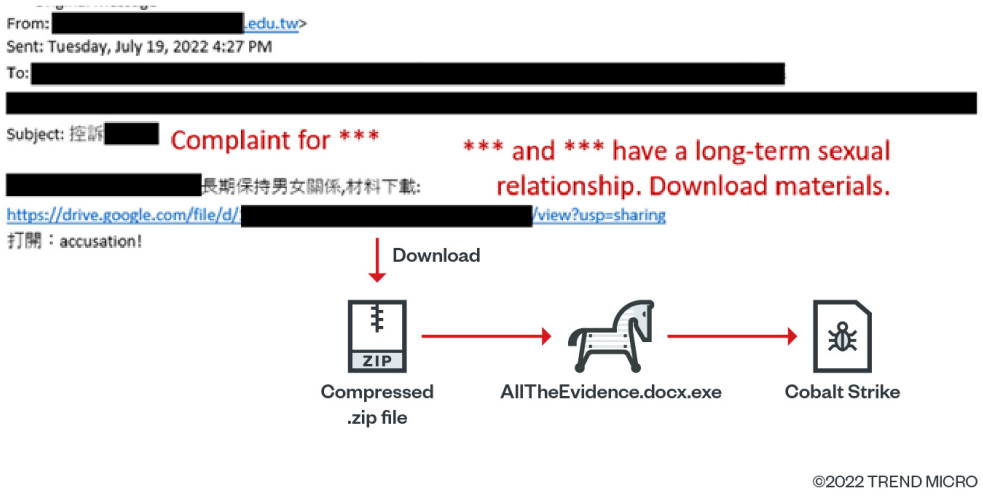

Figure 2. Malware delivery via spear-phishing email in traditional Chinese

In some cases, we also found that the group exploited publicly available applications to deploy and execute a simple downloader to download a shellcode loader and the necessary hack tools for the routine.

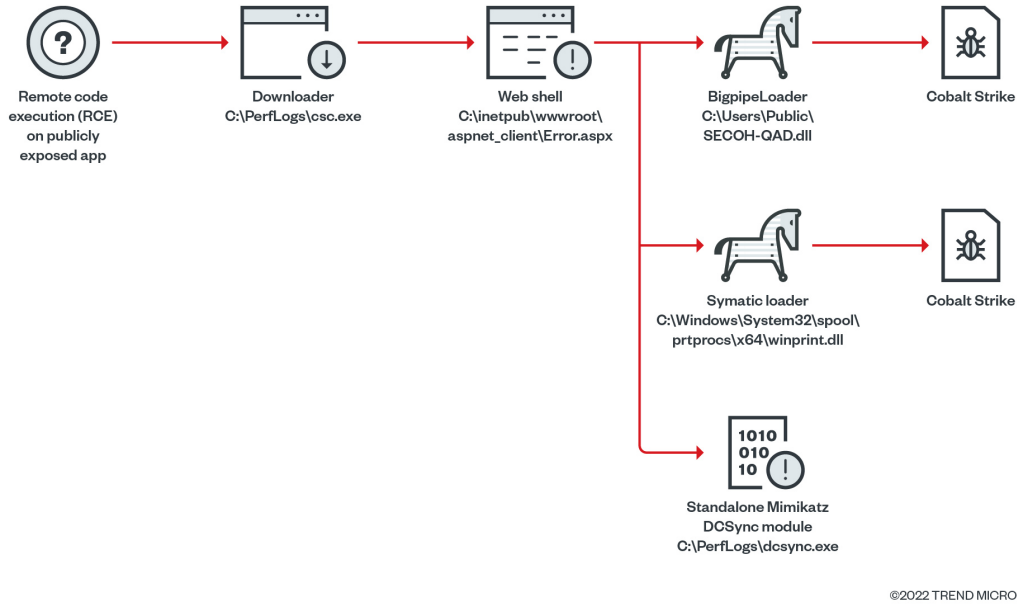

Figure 3. Deliver malware through exploiting exposed applications

Campaign No. 1: May 2020 - Feb 2021

We tracked Earth Longzhi mainly targeting the government, healthcare, academic, and infrastructure industries in Taiwan with a custom Cobalt Strike loader, which we have called Symatic loader, and custom hacking tools.

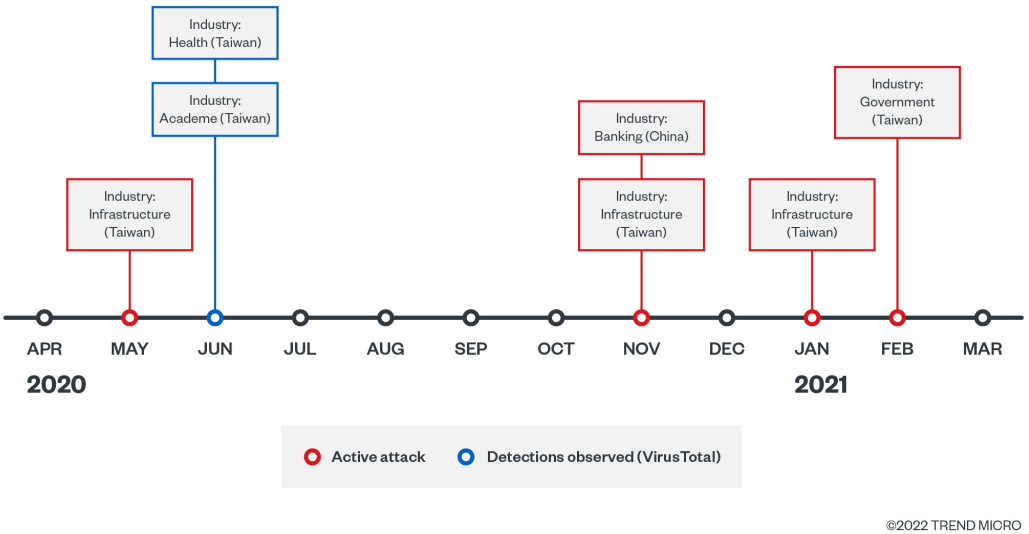

Figure 4. Timeline of attacks during the first campaign

Symatic loader

Symatic is the primary loader used to load the Cobalt Strike payload in the first campaign. To avoid being detected, Symatic adopts the following techniques:

- Restoring in-memory hooks in the user-mode face of the Windows kernel utility *ntdll.dll* by anti-hooking
- Masquerading the parent process by API UpdateProcThreadAttribute
- Injecting a decrypted payload into the system built-in process (*dllhost.exe* or *rundll32.exe*)

Security solutions place the in-memory API hooks in *ntdll.dll* to monitor suspicious behavior. Symatic removes the API hooks first and gets the raw content of *ntdll.dll* from the disk. It then proceeds to replace the in-memory ntdll image to make sure there are no hooks placed in *ntdll.dll*.

```
v1 = GetModuleHandleA("ntdll");
K32GetModuleInformation(v0, v1, &modinfo, 0x18u);// get in-memory ntdll image
pNtdllImageDosHeader = (PIMAGE_DOS_HEADER)modinfo.lpBaseOfDll;
v3 = CreateFileA("C:\\Windows\\System32\\ntdll.dll", 0x80000000, 1u, 0i64, 3u, 0, 0i64);// get raw ntdll from disk
v4 = CreateFileMappingA(v3, 0i64, 0x1000002u, 0, 0, 0i64);
v5 = (char *)MapViewOfFile(v4, 4u, 0, 0, 0i64);
dwSectionIndex = 0;
pNtdllImageNtHeaders = (PIMAGE_NT_HEADERS)((char *)pNtdllImageDosHeader + pNtdllImageDosHeader->e_lfanew);
pNtdllFileBase = v5;
if ( pNtdllImageNtHeaders->FileHeader.NumberOfSections )
{
  do
  {
    v9 = 0i64;
    v10 = (PBYTE)pNtdllImageNtHeaders + 40 * dwSectionIndex + pNtdllImageNtHeaders->FileHeader.SizeOfOptionalHeader;//
                                        // It does not locate to the beginning of IMAGE_SECTION_HEADER directly.
                                        // 40 is sizeof(IMAGE_SECTION_HEADER).
    while ( 1 )
    {
      v11 = v10[v9++ + 24];             // The offset plus 24 = IMAGE_SECTION_HEADER[dwSectionIndex].Name
      if ( v11 != aText[v9 - 1] )       // find .text section
        break;
      if ( v9 == 6 )
      {
        dwVirtualSize = *((unsigned int *)v10 + 8);// IMAGE_SECTION_HEADER.VirtualSize
        pVirtualAddress = (char *)pNtdllImageDosHeader + *((unsigned int *)v10 + 9);// IMAGE_SECTION_HEADER.VirtualAddress
        flOldProtect = 0;
        VirtualProtect(pVirtualAddress, dwVirtualSize, 0x40u, &flOldProtect);
        memmove(
          (char *)pNtdllImageDosHeader + *((unsigned int *)v10 + 9),
          &pNtdllFileBase[*((unsigned int *)v10 + 9)],
          *((unsigned int *)v10 + 8));  // Copy raw ntdll mapping from disk to memory
        VirtualProtect(
          (char *)pNtdllImageDosHeader + *((unsigned int *)v10 + 9),
          *((unsigned int *)v10 + 8),
          flOldProtect,
          &flOldProtect);
        break;
      }
    }
    ++dwSectionIndex;
  }
  while ( dwSectionIndex < pNtdllImageNtHeaders->FileHeader.NumberOfSections );
}
```

Figure 5. Symatic Loader's detection evasion techniques

After restoring the ntdll, Symatic will spawn a new process for process injection. It is worth noting that it will masquerade the parent process of the newly created process to obfuscate the process chain.

```
if ( v24 == 7 )
{
  v26 = GetProcessIdByName((__int64)"svchost.exe");
  Value = OpenProcess(0x1FFFFFu, 0, v26);
  memset(&StartupInfo.StartupInfo.lpReserved, 0, 0x68ui64);
  StartupInfo.StartupInfo.cb = 112;
  *(_QWORD *)pcbBuffer = 0i64;
  ProcessInformation.hProcess = 0i64;
  ProcessInformation.hThread = 0i64;
  *(_QWORD *)&ProcessInformation.dwProcessId = 0i64;
  InitializeProcThreadAttributeList(0i64, 1u, 0, (PSIZE_T)pcbBuffer);
  v27 = (struct _PROC_THREAD_ATTRIBUTE_LIST *)LocalAlloc(0x40u, *(SIZE_T *)pcbBuffer);
  InitializeProcThreadAttributeList(v27, 1u, 0, (PSIZE_T)pcbBuffer);
  if ( !UpdateProcThreadAttribute(v27, 0, PROC_THREAD_ATTRIBUTE_INPUT, &Value, 8ui64, 0i64, 0i64) )// masquerade the parent process as svchost.exe
    return 1;
  StartupInfo.lpAttributeList = v27;
  if ( !CreateProcessAsUserA(
          0i64,
          0i64,
          (LPSTR)"C:\\Windows\\System32\\dllhost.exe",
          0i64,
          0i64,
          0,
          0x80000u,
          0i64,
          "C:\\Windows\\System32",
          &StartupInfo.StartupInfo,
          &ProcessInformation) )
    return 1;
```

Figure 6. Obfuscating the process chain

All-in-one hack tool

For the post-exploitation operations of this campaign, Earth Longzhi also prepares an all-in-one tool to combine all the necessary tools in one package. Most of the tools included in this one package are either publicly available or were used in previous attack deployments. This compressed tool allows them to complete multiple operations by using a single executable in their operation.

Table 1. All the tools needed for the routine in one executable

Arguments	**Function**
-P	HTRan
-S	Socks5 proxy
-SQL	Password scans against Microsoft SQL server (MSSQL) with a given dictionary
-IPC	Password scans over $IPC with a given dictionary
-SFC	Disables Windows File Protection via *SFC_OS.dll*
-filetime	Modifies a specific file's timestamp
-Port	TCP (Transmission Control Protocol) port scanner
-Runas	Launches a process with higher privileges
-Clone	Clones specified users' relative ID (RID) in registry for RID spoofing
-driver	Gets information of local or remote drives (using NetShareEnum)
-sqlcmd	Command will be executed with SQLExecDirect

```
====================== HD All in One Tool V2.00 (2014-09-01) =================
================ Code by William Henry, Thanks for Steve Paul Jobs============

[Usage of Function:]
        -p              Packet Transmit
        -S              Socks5 Proxy
        -SQL            MSSql Password Scanner
        -IPC            IPC$ Password Scanner
        -SFC            DisableSFC
        -filetime       Chanage File Time
        -Port           Port Scan
        -Runas          Run as
        -Clone          Clone User
        -driver         Get Driver Space
        -Sqlcmd         SqlServer Cmd
```

Figure 7. All-in-one tool available since 2014

Second campaign: August 2021 to June 2022

Earth Longzhi initiated the second campaign five months after the last attack in its first campaign. In this campaign, the APT group used various types of customized Cobalt Strike loaders, which we call CroxLoader, BigpipeLoader, and OutLoader. We also found other customized hacking tools.

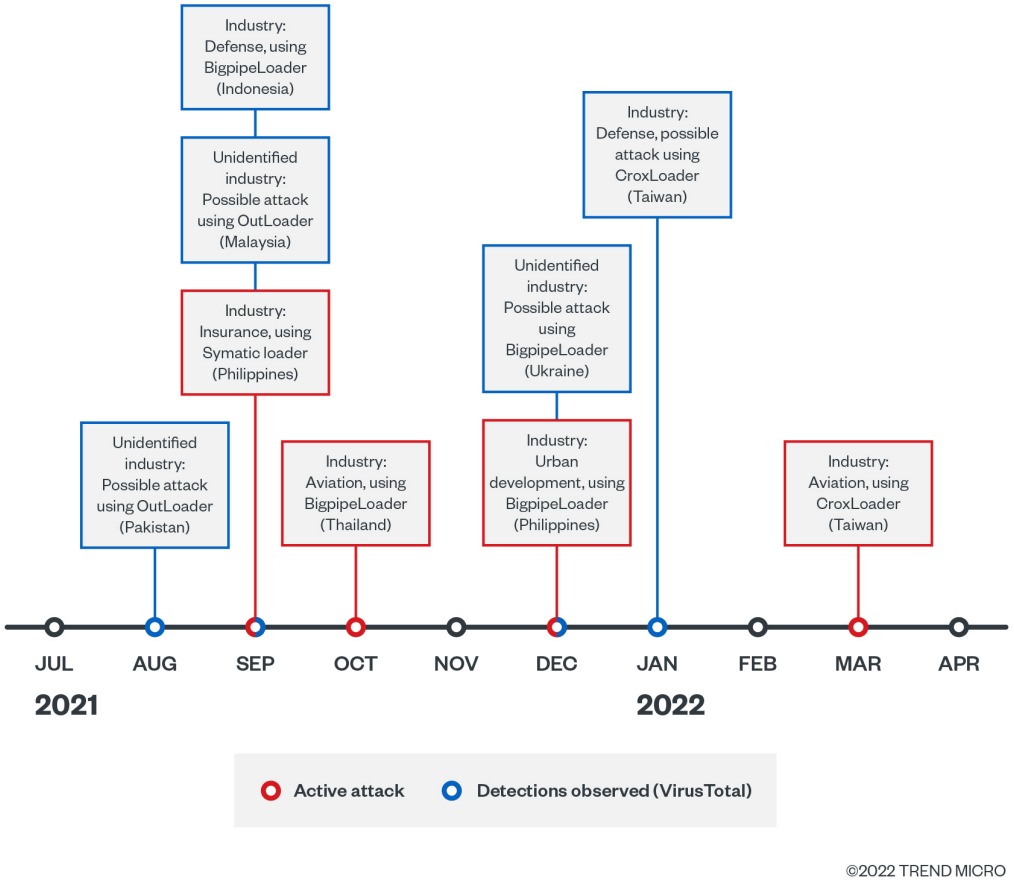

Figure 8. Timeline of attacks during the second campaign

Custom loaders

We discovered several custom loaders of Cobalt Strike, including similar samples uploaded in VirusTotal. Each loader implemented a different algorithm to decrypt the payload, as follows:

Table 2. Summary of customized loaders in the second campaign

Name	Observed	Algorithm	Extra feature
CroxLoader	Oct 2021 onward	• XOR 0xCC + SUB 0xA • RtlDecompressBuffer + XOR 0xCC	• Process injection • Decoy document
BigpipeLoader	Aug 2021 onward	• Base64 + RSA + AES128-CFB • AES128-CFB	• Multi-threading decryption over named pipe • Decoy document
MultiPipeLoader	Aug 2021	Base64 + AES128-CFB	• Multi-threading decryption over named pipe • Decoy document
OutLoader	Sep 2021	AES128-CFB	• Downloads payload from an external server • Decoy document

CroxLoader

During the deployment of the second campaign, we found two different variants of CroxLoader with respective patterns of use. The first variant is commonly used when attackers use publicly facing applications as the entry point of attack. It decrypts the embedded payload and injects the decrypted payload into the remote process. Meanwhile, the second variant of CroxLoader is often deployed through spearphishing emails to lure victims into opening it. The variant used for each targeted victim depends on the applicable attack scenario.

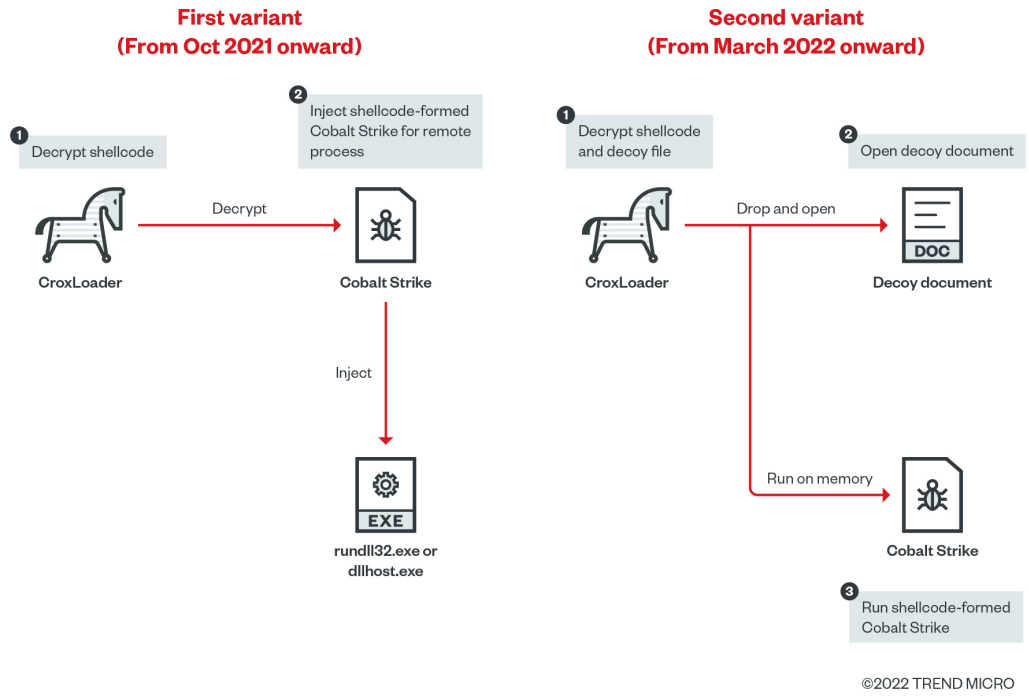

Figure 9. TTPs of the CroxLoader variants

BigpipeLoader

Since this loader will read/write encrypted payload through a named pipe, we named this shellcode loader BigpipeLoader. In one of our threat hunting sessions, we found two variants of this loader with different execution procedures. The first variant of BigpipeLoader just drops the decoy file and loads the Cobalt Strike payload into the memory, then proceeds to execute it. In the second variant, however, the attacker creates a dropper, which drops the malicious *WTSAPI32.dll* designed to be sideloaded by a legitimate application with the file name "*wusa.exe*". This launches the encrypted BigpipeLoader (*chrome.inf*). Both variants of BigpipeLoader use the AES-128-CFB algorithm to decrypt the payload.

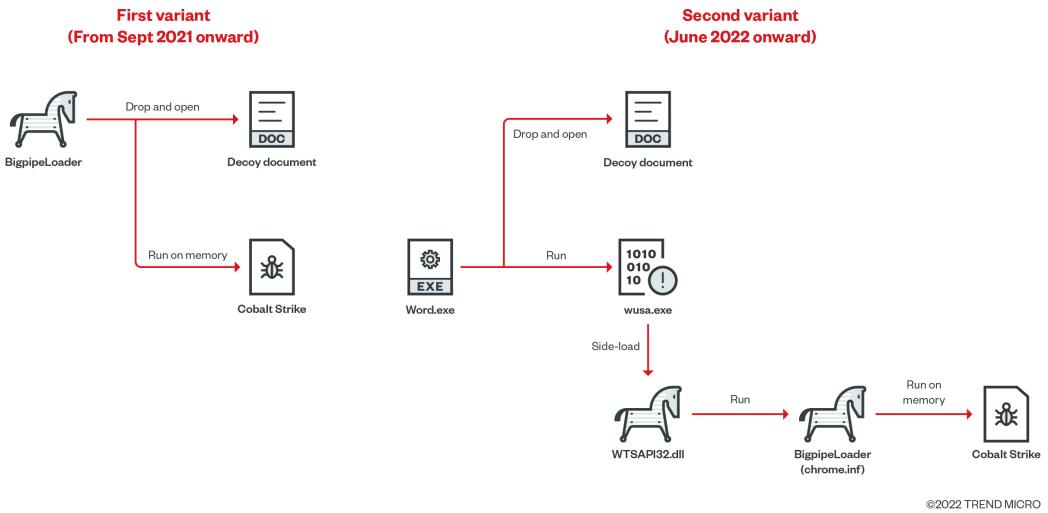

Figure 10. TTPs of the BigpipeLoader variants

Meanwhile, MultipipeLoader and OutLoader are similar to CroxLoader and BigpipeLoader but have slightly different features. MultipipeLoader uses multiple threads to read/write the encrypted payload like BigpipeLoader, but it implements a similar decryption routine as CroxLoader. Meanwhile, OutLoader tries to download the payload from a remote server, while its other function is the same as BigpipeLoader. From these minimal variations, we believe the attacker is trying to develop new loaders by combining existing features of other, previously used loaders.

Post-exploitation

During the investigation of the second campaign, we collected multiple hacking tools used for privilege escalation (PrintNightmare and PrintSpoofer), credential dumping (custom standalone Mimikatznews- cybercrime-and-digital-threats), and defense evasion (disablement of security products). Instead of using public tools as they are, the threat actors are able to reimplement or develop their own tools based on some open-source projects. In the following subsections, we introduce these hack tools.

Custom standalone Mimikatz

Earth Longzhi reimplemented some modules of Mimikatz (shown in Table 3) as standalone binaries. Upon comparing the binary and source code, the attacker just removed the necessary code snippet from the public code and compiled it as standalone binary. We call this technique "Bring-Your-Own Mimikatz." The reimplementation of open-source hacking tools such as Mimikatz is common among red-team community groups for reducing chances of detection.

We also observed the standalone version of the *sekurlsa::logonpasswords* module, which abuses the vulnerable driver *RTCore64.sys* to disable the Protected Process Light (PPL) mechanism to dump credentials from *lsass.exe*. We will introduce how this vulnerable driver helps to bypass the PPL.

Table 3. Reimplemented Mimikatz modules and their functions

Reimplemented Mimikatz modules	**Description of reimplemented function**
sekurlsa::logonpasswords	To dump credentials from *lsass.exe*; some variants support disabling PPL by using the vulnerable driver.
lsadump::dcsync	To perform a DCSync attack
lsadump::backupkeys + dpapi::chrome	To combine two different modules to retrieve a backup key from domain controller (DC) and use the key to decrypt chrome’s credential data protected by Data Protection API (DPAPI)
misc::memssp	To dump credentials through Security Support Provider (SSP); implemented based on @XPN

Security product disablement

For disabling security products, we found two different tools, which we named ProcBurner and AVBurner. Both tools abuse the vulnerable driver (*RTCore64.sys*) to modify the specified value in the kernel object. *RTCore64.sys* is a component of Afterburner. In 2019, this driver was assigned as CVE-2019-16098, which allows authenticated users to read/write any arbitrary address including kernel space. However, the outdated version of vulnerable driver still has a valid signature. As a result, the attacker can deliver the outdated version of the driver into the victim machine and abuse it for various purposes, such as for anti-antivirus or anti-EDR. This technique is known as "Bring-Your-Own Vulnerable Driver."

Figure 11. CVE-2019-16098 in RTCore64.sys

ProcBurner is designed to terminate specific running processes. Simply put, it tries to change the protection of the target process by forcibly patching the access permission in the kernel space using the vulnerable *RTCore64.sys*. We show the workflow of ProcBurner here (note that the environment used is Windows 10 20H2 x64):

1. OpenProcess with PROCESS_QUERY_LIMITED_INFORMATION (=0x1000).
2. Return HANDLE of target process (0x1d8).
3. Get the address of HANDLE_TABLE_ENTRY object of target handle by tracking back from EPROCESS object.
4. Send IOCTL request to mask HANDLE_TABLE_ENTRY. GrantedAccessBits of target process with PROCESS_ALL_ACCESS (=0x1fffff).
5. Vulnerable *RTCore64.sys* writes the requested bitmask value.
6. Terminate process.

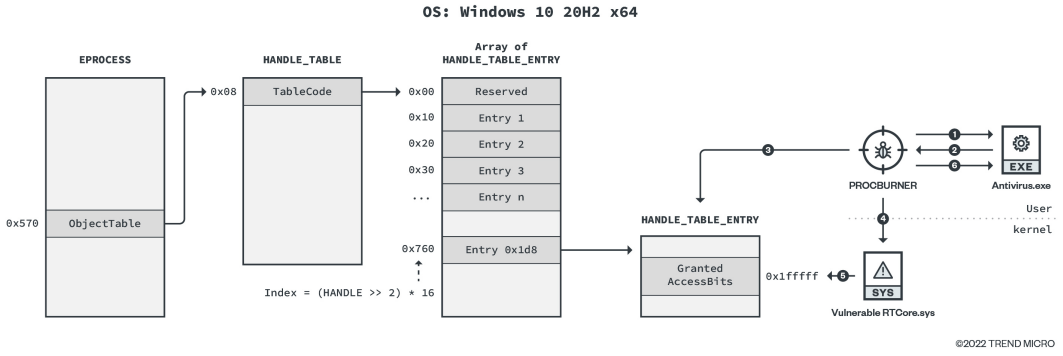

Figure 12. The workflow of ProcBurner

Specific to ProcBurner, it can check the currently running operating system version before patching. ProcBurner hard-codes the offset of kernel objects' field, which can be different for each build version. If ProcBurner supports the offset correctly, it should work on any of the versions listed. The following versions are supported:

- Windows 7 SP1
- Windows Server 2008 R2 SP1
- Windows 8.1
- Windows Server 2012 R2
- Windows 10 1607
- Windows 10 1809
- Windows Server 2018 1809
- Windows 10 20H2
- Windows 10 21H1
- Windows 11 21H2
- Windows 11 22449
- Windows 11 22523
- Windows 11 22557

For AVBurner, this tool is designed for removing the kernel callback routine to unregister the AV/EDR product. To understand how AVBurner works, we will briefly introduce kernel callback.

Kernel callback is a Windows OS mechanism to allow drivers, including antivirus drivers, to register callback routines to receive notifications on certain events such as process, thread, or registry creation. *Ntoskrnl.exe* provides several APIs for drivers to register callbacks for each event. For example, for monitoring process creation, PsSetCreateProcessNotifyRoutine is exported. This API receives the function pointer to invoke when any process is created. When PsSetCreateProcessNotifyRoutine is called, it invokes PspSetCreateProcessNotifyRoutine. In this function, Windows kernel registers the given callback function at the end of a callback array named PspCreateProcessNotifyRoutine. After this, when any process is created, Windows kernel enumerates this table to find the callback function.

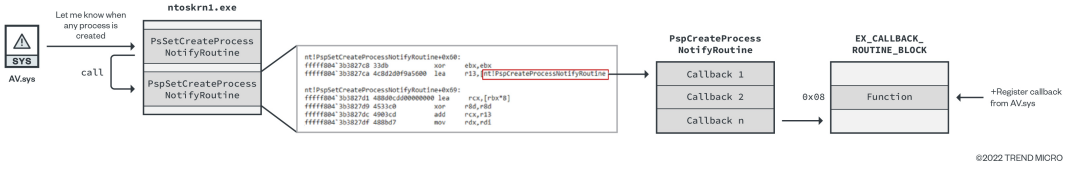

Figure 13. AV.sys registers callback for process creation event by calling the PsSetCreateProcessNotifyRoutine API

AVBurner abuses *RTCore64.sys* to enumerate the PspCreateProcessNotifyRoutine array to find the target driver. The workflow of AVBurner is as follows:

1. Get addresses of PsSetCreateProcessNotifyRoutine and IoCreateDriver.
2. Search for a specific sequence of bytes to find the address of PspCreateProcessNotifyRoutine between the above addresses (PsSetCreateProcessNotifyRoutine and IoCreateDriver).

3. PspCreateProcessNotifyRoutine is a table of callback functions that contains the custom pointer to object EX_CALLBACK_ROUTINE_BLOCK. The address of the said object can be calculated by removing the last four bits of the pointer.
4. EX_CALLBACK_ROUTINE_BLOCK.Function (offset=0x08) contains a pointer to the callback function (Driver.sys in this case). Get the driver's file path that the callback function belongs to, and if the driver's file property has target string (such as Trend), AVBURNER overwrites the pointer with NULL, resulting in the removal of the callback registration.

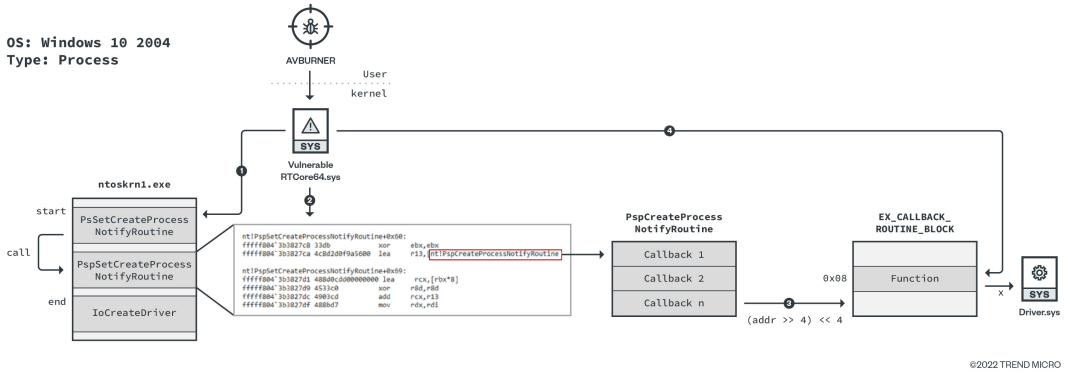

Figure 14. The workflow of AVBurner

Attribution

We attributed these threat actors to APT41's subgroup Earth Longzhi based on the following factors.

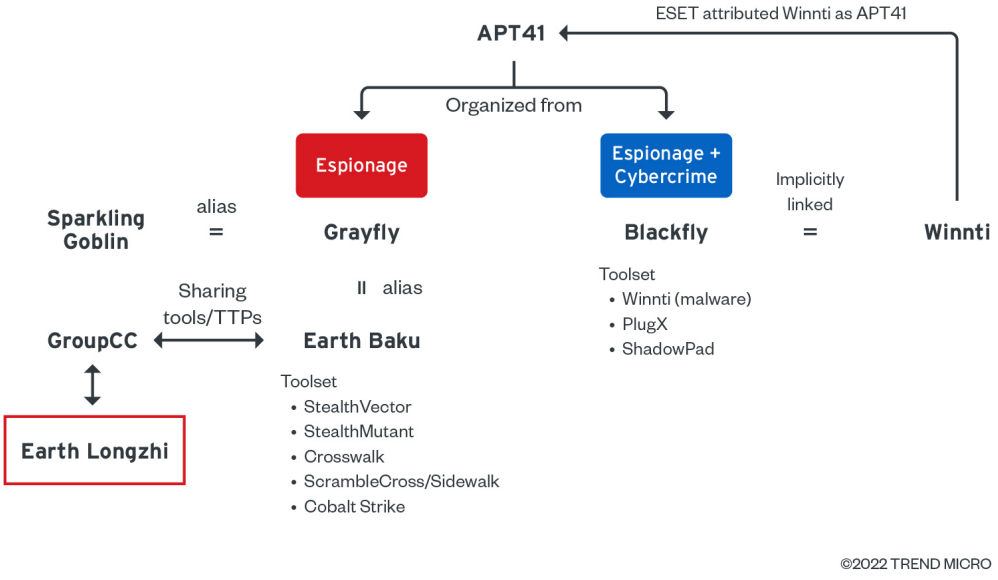

Figure 15. Finding Earth Longzhi's position in the APT41 organizational structure

Victimology

The affected regions and targeted sectors are countries of interest located in the East and Southeast Asia, which is close to the victimology identified in our research on another APT41 subgroup, Earth Baku.

Shared Cobalt Strike metadata with other APT41 subgroups

After checking all the metadata of the Cobalt Strike payloads, we found that most of payloads shared the same watermark, *426352781*, and public key *9ee3e0425ade426af0cb07094aa29ebc*. This watermark and public key combination is also used by Earth Baku and GroupCC, which are also believed to be subgroups of APT41. The identified watermark has not yet been attributed to other threat actors. The use of the same watermark and public key indicates Earth Longzhi sharesing the Cobalt Strike team server, as well as Cobalt Strike package and license with the other APT41 subgroups.

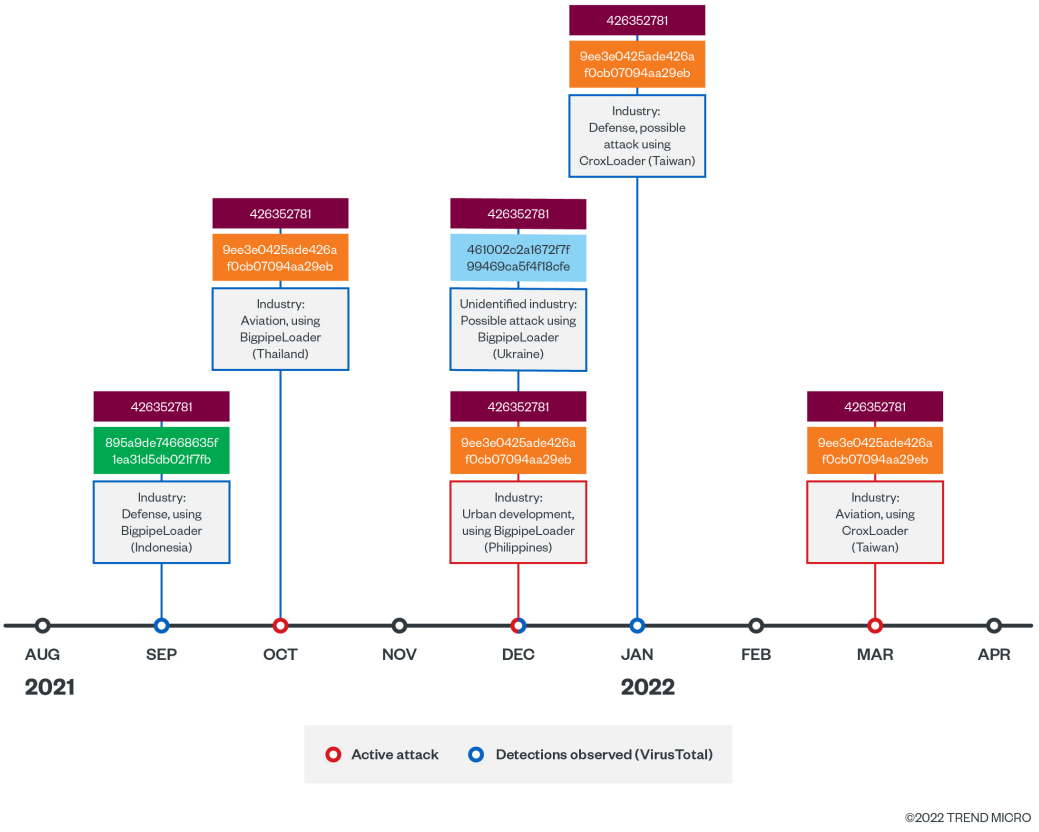

Figure 16. Timeline of attacks with shared Cobalt Strike metadata

Code similarities of shellcode loaders and overlapping TTPs

We also found that the decryption algorithms in Symatic Loader and CroxLoader are quite similar to the one identified with GroupCC. All of the said loaders use *<(sub 0xA) XOR 0xCC>* as their decryption algorithm. As for the similar TTPs, Earth Longzhi also adopted the Python Fastly CDN used by GroupCC to hide the actual command-and-control (C&C) server address. At the time we were analyzing Earth Longzhi, we did not find reports documenting the abuse of Python CDN, other than the GroupCC report by Team T5. Hence, we consider it as evidence of the relationship between Earth Longzhi and GroupCC.

Figure 17. Decryption algorithm used by GroupCC (top), CroxLoader (left), and Symatic loader (bottom)

Conclusion

We profile Earth Longzhi as an APT group that mainly targets the Asia-Pacific region. After investigating two different campaigns, we verified that its target sectors are in industries pertinent to Asia-Pacific countries' national security and economies. The activities in these campaigns show that the group is knowledgeable on red-team operations. The group uses social engineering techniques to spread its malware and deploy customized hack tools to bypass the protection of security products and steal sensitive data from compromised machines. From an overall security perspective, it seems that Earth Longzhi is playing Hack The Box, an online platform for penetration testing, but in the real world.

APT41 groups are seemingly using less custom malware but are getting more accustomed to using more commodity malware such as Cobalt Strike. They are also now more focused on developing new loaders and hacking tools to bypass security products. AVBurner is a formidable example of this, as it disables solutions that still use the dated and vulnerable driver, while both ProcBurner and AVBurner focus on kernel-level security — a noticeable emerging pattern among APT groups and cybercrime. In addition, Earth Longzhi, as a subgroup of APT41, appears familiar with offensive security teams such as red teams.

In the process of attribution, we also discovered that the group uses shared Cobalt Strike licenses and imitates the TTPs used with other APT41 subgroups. The behavior of sharing tools between different groups could point to the following circumstances:

1. These threat actors are no longer static groups. Although the organizational structure will keep changing from time to time, the tools will be inherited by the subsequent newly organized groups.
2. The tool developers and campaign operators share the tools with their collaborator groups.

Following these indications, tool-based attribution and analysis will likely become more complicated and will be a challenge to threat researchers in figuring links among different groups. Researchers of APT groups and other cybercriminals will also have to consider other aspects and integrate collected information such as code similarities and victim profiles, among other technical characteristics for consideration. Security providers and solutions will also have to reassess and, if possible, avoid or disable the use of vulnerable drivers. At the very least, organizations' security teams should be allowed to enable features such as monitoring of vulnerable driver installation, if available. Fortunately for researchers and operational security teams, these groups' use of publicly available tools and previously deployed routines can be detected faster and can be tested using their TTPs.

Indicators of Compromise (IOCs)

Find the full list of IOCs hereopen on a new tab.

MITRE ATT&CK

Initial Access	Reconnaissance	Defense Evasion	Persistence	Privilege Escalation	Credential Access	Command and Control
T1566.001 Spearphishing Attachment **T1190** Exploit Public-Facing Application	**T1589.002** Email Addresses	**T1574.002** Hijack Execution Flow: DLL Side-Loading **T1055** Process Injection **T1211** Exploitation for Defense Evasion **T1562.001** Disable or Modify Tools **T1036.007** Double File Extension **T1070.006** Timestomp	**T1547.012** Boot or Logon Autostart Execution: Print Processors	**T1068** Exploitation for Privilege Escalation	**T1555.003** Credentials from Web Browsers **T1003.001** LSASS Memory **T1003.006** DCSync	**T1071.001** Application Layer Protocol: Web Protocols **T1090.001** Internal Proxy **T1090.002** External Proxy **T1090.004** Domain Fronting **T1095** Non-Application Layer Protocol **T1573.002** Encrypted Channel: Asymmetric Cryptography

Tags

Source: https://www.trendmicro.com/en_us/research/22/k/hack-the-real-box-apt41-new-subgroup-earth-longzhi.html