New Hancitor Malware: Pimp my Downloaded

blog.minerva-labs.com/new-hancitor-pimp-my-downloader

- Tweet
-

Hancitor (AKA Chanitor and TorDal) is a downloader-type malware – out there for almost two years now. Downloaders contact the C2 servers after establishing an initial foothold on the victim's machine – downloading and installing Trojans, bots and other kinds of malware.

Last May malware researchers at Proofpoint revealed that they observed the re-emergence of Hancitor.

This specific downloader has three core capabilities:

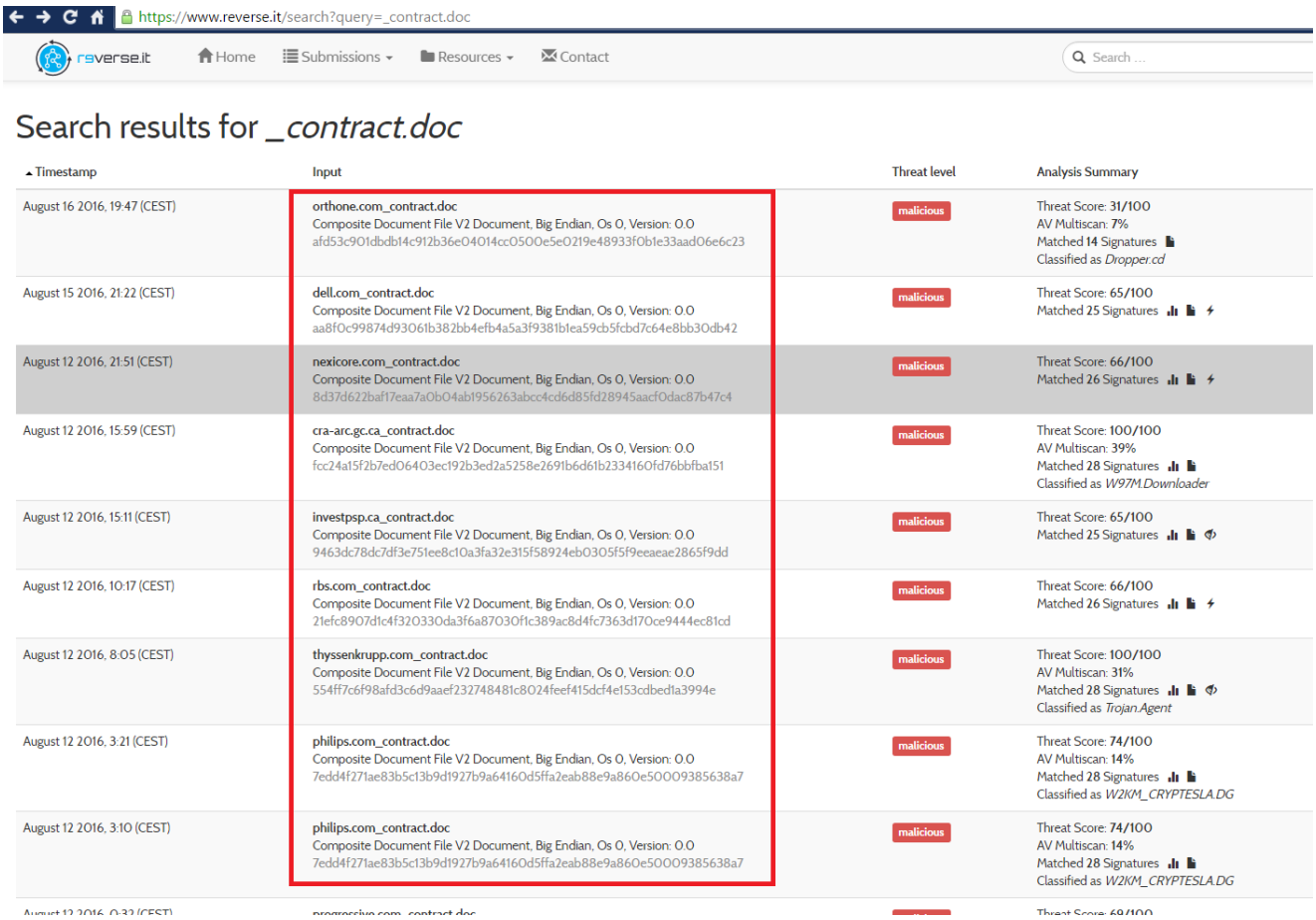
- Downloading and executing an exe file from a URL
- Downloading a DLL from a URL and executing it without writing it to the disk, but by writing it directly to the memory space of the downloader
- Deleting itself

Any of those commands may be received by the downloader after transmitting a "beacon" HTTP post request to the C2 server. This request includes basic fingerprinting info unique to each endpoint and enables the attacker to easily manage the machines of many victims concurrently while possibly infecting different endpoints with different types of malware in later infection stages.

The Augmented Hancitor

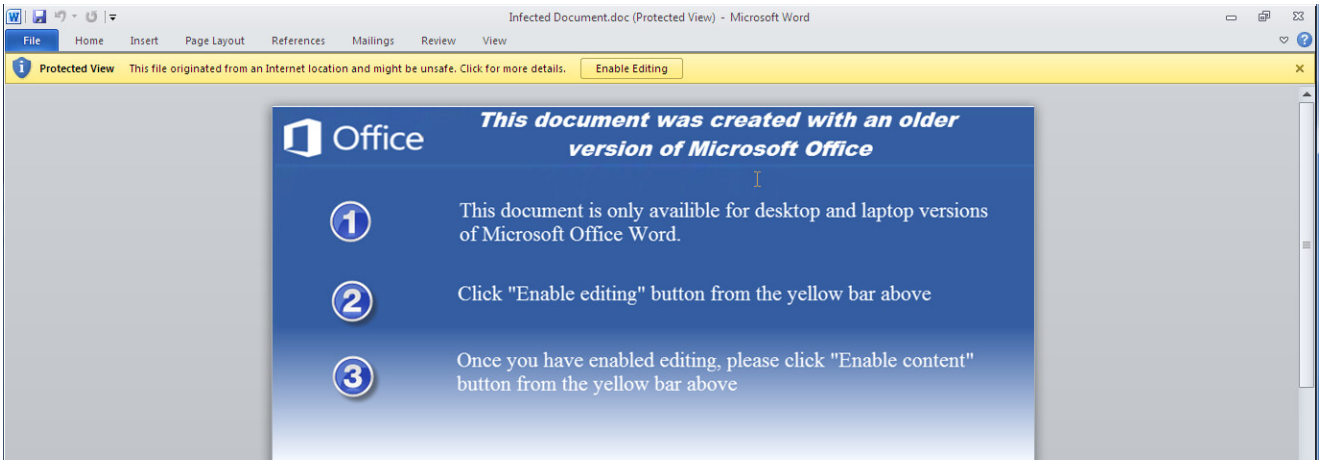
Last week we were contacted by one of our clients after he received a notification from one of Minerva's agents.

A short forensic analysis enabled us to trace a phishing email containing a malicious attachment titled *CompanyPublicMailServer.com_contract*. We assumed that this is a wide Dridex-style spam based infection campaign and indeed a simple search in a publicly available sandbox proved that this was a pattern as we were able to find over 20 different malicious documents similar to the one sent to our clients:



Documents infected with Hancitor

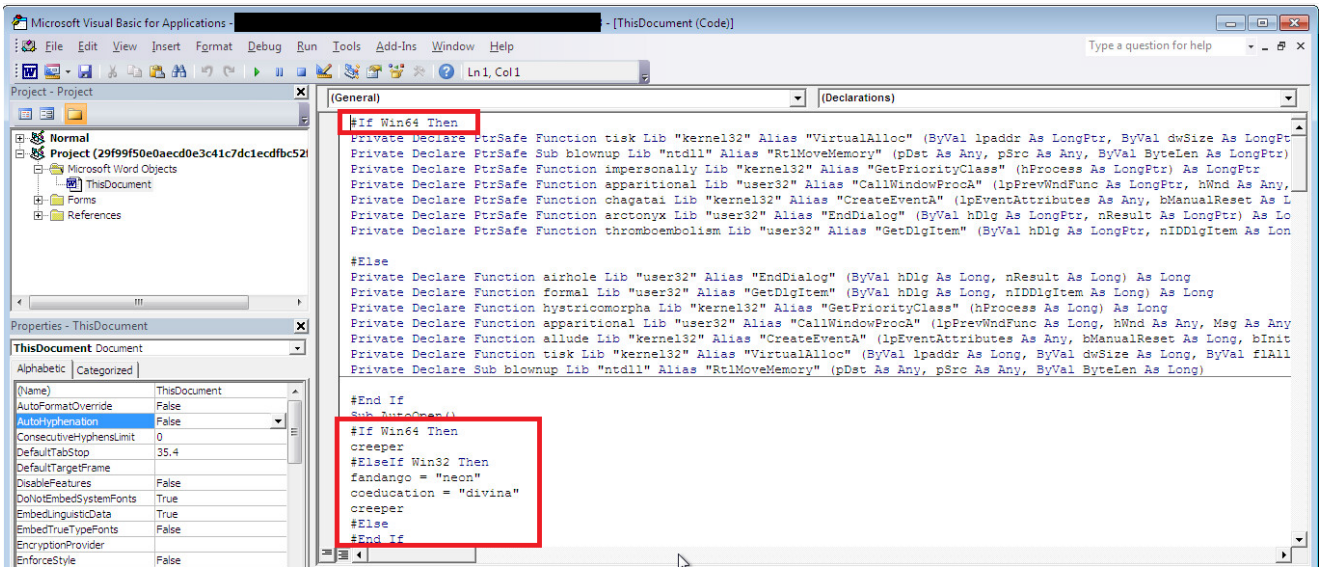
The malicious Microsoft Word *.doc* attachment had an embedded VBA macro script with a short message aimed at luring the victim to enable the execution of the script.



After enable editing is clicked - malware will execute

Unlike the document used to drop Hancitor in Proofpoint's investigations our sample had some extra features:

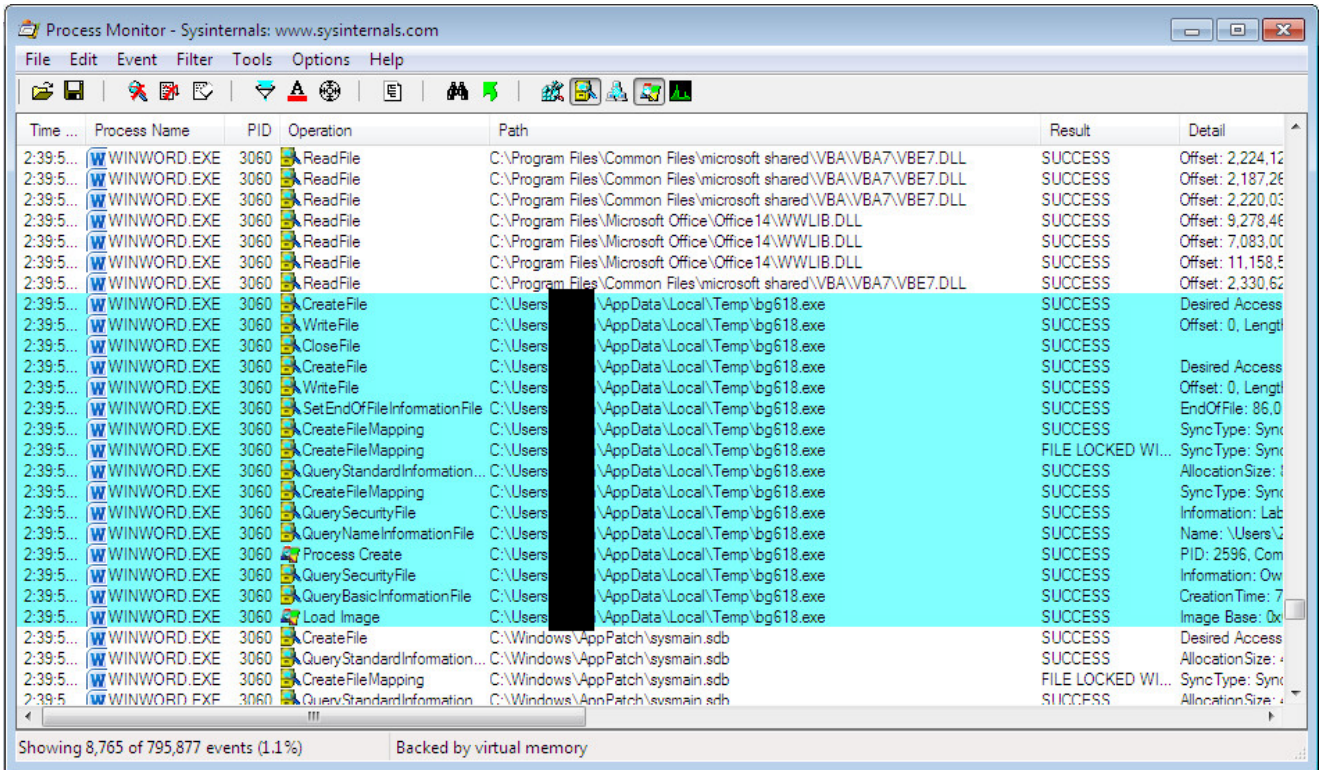
- **Handling x86/x64 architectures seamlessly** – including adaptation of pointers and imported functions:



Changing the flow according to the OS version

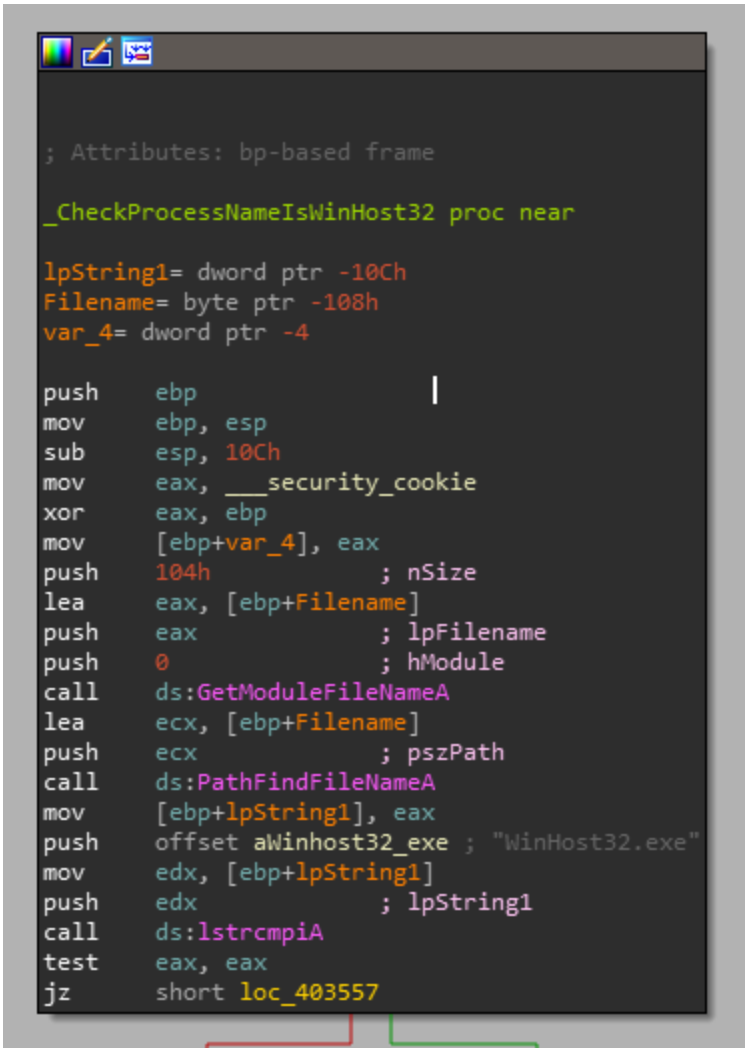
These characteristics greatly increase the chances of successfully infecting the victim's machine, saving noisy crashes of the macro as a bonus.

> **Using *CallWindowProcA* Windows API to execute a code written to the heap** – As explained in Waleed Assar's blog – it allows the malware to avoid suspicious API calls as *ShellExecute* and *CreateProcess* and the need to write this intermediate shellcode-like dropper stage to the disk. It is uncommon to see this technique implemented in VBA script, however – it is used by .exe files in the wild at least since *Citadel*.



Hancitor dropped and executed by the malicious macro

Now, Hancitor is finally running. On its initial execution it is running under a random looking hard coded name (we observed *bg618.exe* and *lj016.exe*) from the %TEMP% folder. It then creates another instance of itself and uses process hollowing to unpack itself to its new instance. The unpacked executable copies Hancitor to either the system or temporary folder under the name *WinHost32.exe*Hancitor then executes the binary under its new name and deletes the old one. It is also taking care of achieving persistency by creating a registry value under *HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run*. From now on, each time it will be executed under the new name the following mechanism will kick in:



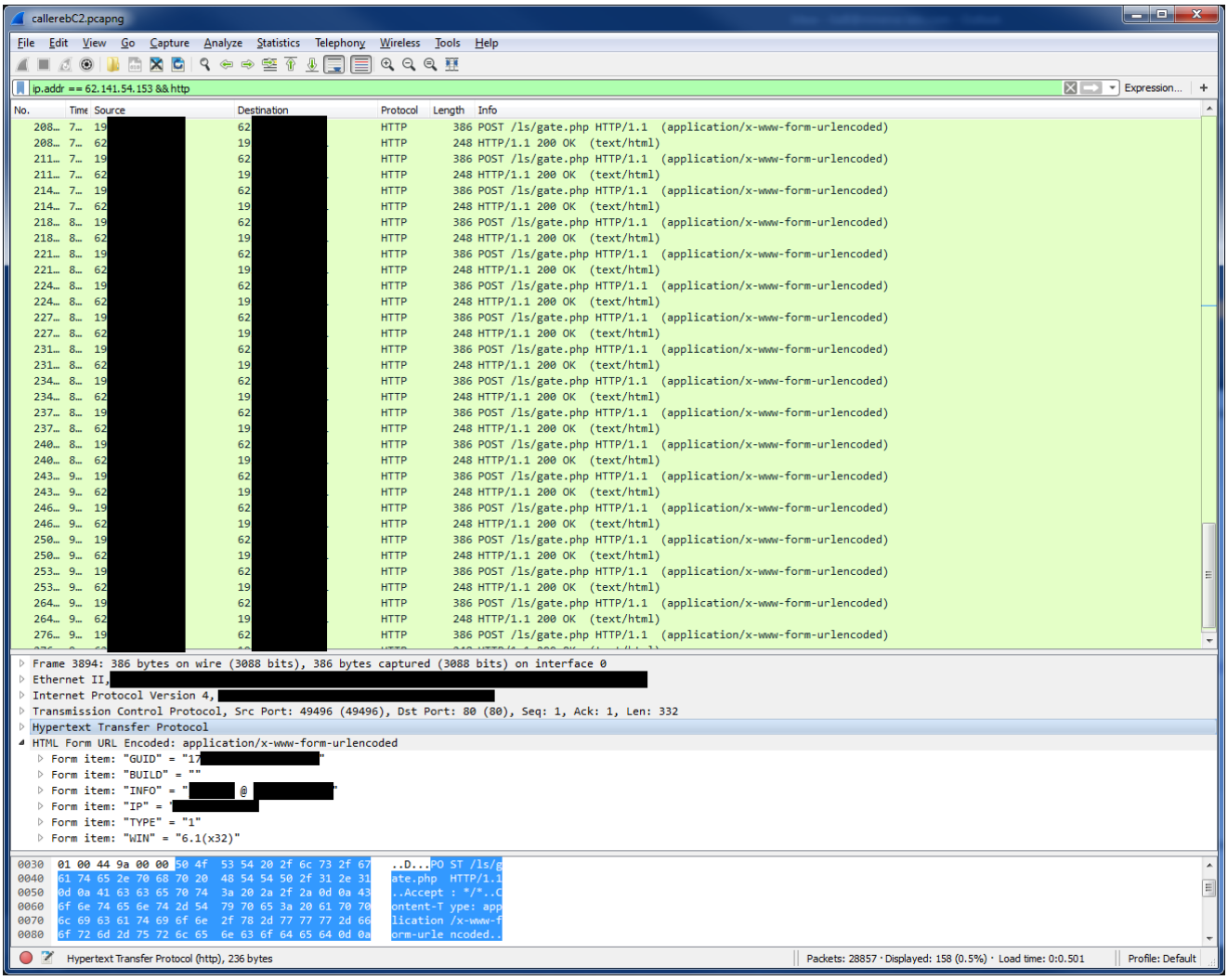
```
; Attributes: bp-based frame

_CheckProcessNameIsWinHost32 proc near

lpString1= dword ptr -10Ch
Filename= byte ptr -108h
var_4= dword ptr -4

push    ebp
mov     ebp, esp
sub     esp, 10Ch
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
push    104h            ; nSize
lea     eax, [ebp+Filename]
push    eax             ; lpFilename
push    0               ; hModule
call    ds:GetModuleFileNameA
lea     ecx, [ebp+Filename]
push    ecx             ; pszPath
call    ds:PathFindFileNameA
mov     [ebp+lpString1], eax
push    offset aWinhost32_exe ; "WinHost32.exe"
mov     edx, [ebp+lpString1]
push    edx             ; lpString1
call    ds:lstrcmpiA
test    eax, eax
jz      short loc_403557
```

This is a simple test that determines if the file is executed for the first time and will gain persistency, or if it is already installed and should initiate its core functionality as a downloader - Hancitor now connects back to its C2 servers in order to download and execute malware. The communication with the C2 server was similar to the pattern described in Proofpoint's report. A "beacon" signal was sent with unique identifiers of the victim:



Hanictor checks if the C2 has new tasks for it

Just as we saw in the "old" Hancitor, our new version is able to receive commands to download and execute malware. We compared the binaries, trying to figure out if there are any changes between Proofpoint's Hancitor and ours and we found one key difference: **Support for a new command-type was added – "*b*".**

```
signed int __cdecl SwitchTable(char *a1, _DWORD *a2)
{
  signed int result; // eax@2

  if ( a1[1] != 58 )
    return 0;
  switch ( *a1 )
  {
    case 'b':
      *a2 = InjectionToSVCHostFromUrl((int)(a1 + 2));
      result = 1;
      break;
    case 'd':
      *a2 = IndirectDeleteSelf();
      result = 1;
      break;
    case 'l':
      *a2 = executeUrlToThread(a1 + 2);
      result = 1;
      break;
    case 'n':
      *a2 = 1;
      result = 1;
      break;
    case 'r':
      *a2 = DownloadToTempAndExecuteFromUrl(a1 + 2);
      result = 1;
      break;
    default:
      result = 0;
      break;
  }
  return result;
}
```

C2 commands switch table

Reverse engineering the function that handles it led us to the conclusion that it is used to execute code downloaded from a URL. However, instead of simply executing it or writing it to Hancitor's memory space it injects it to a svchost.exe process

```
push    offset Name     ; "SystemRoot"
call    ds:GetEnvironmentVariableA
push    offset aSystem32Svchos ; "\\System32\\svchost.exe"
lea     edx, [ebp+Buffer]
push    edx             ; lpString1
call    ds:lstrcatA
lea     eax, [ebp+ProcessInformation]
push    eax             ; lpProcessInformation
lea     ecx, [ebp+StartupInfo]
push    ecx             ; lpStartupInfo
push    0               ; lpCurrentDirectory
push    0               ; lpEnvironment
push    424h            ; dwCreationFlags
push    0               ; bInheritHandles
push    0               ; lpThreadAttributes
push    0               ; lpProcessAttributes
lea     edx, [ebp+Buffer]
push    edx             ; lpCommandLine
push    0               ; lpApplicationName
call    ds:CreateProcessA
test    eax, eax
jnz     short loc_402D5D
```

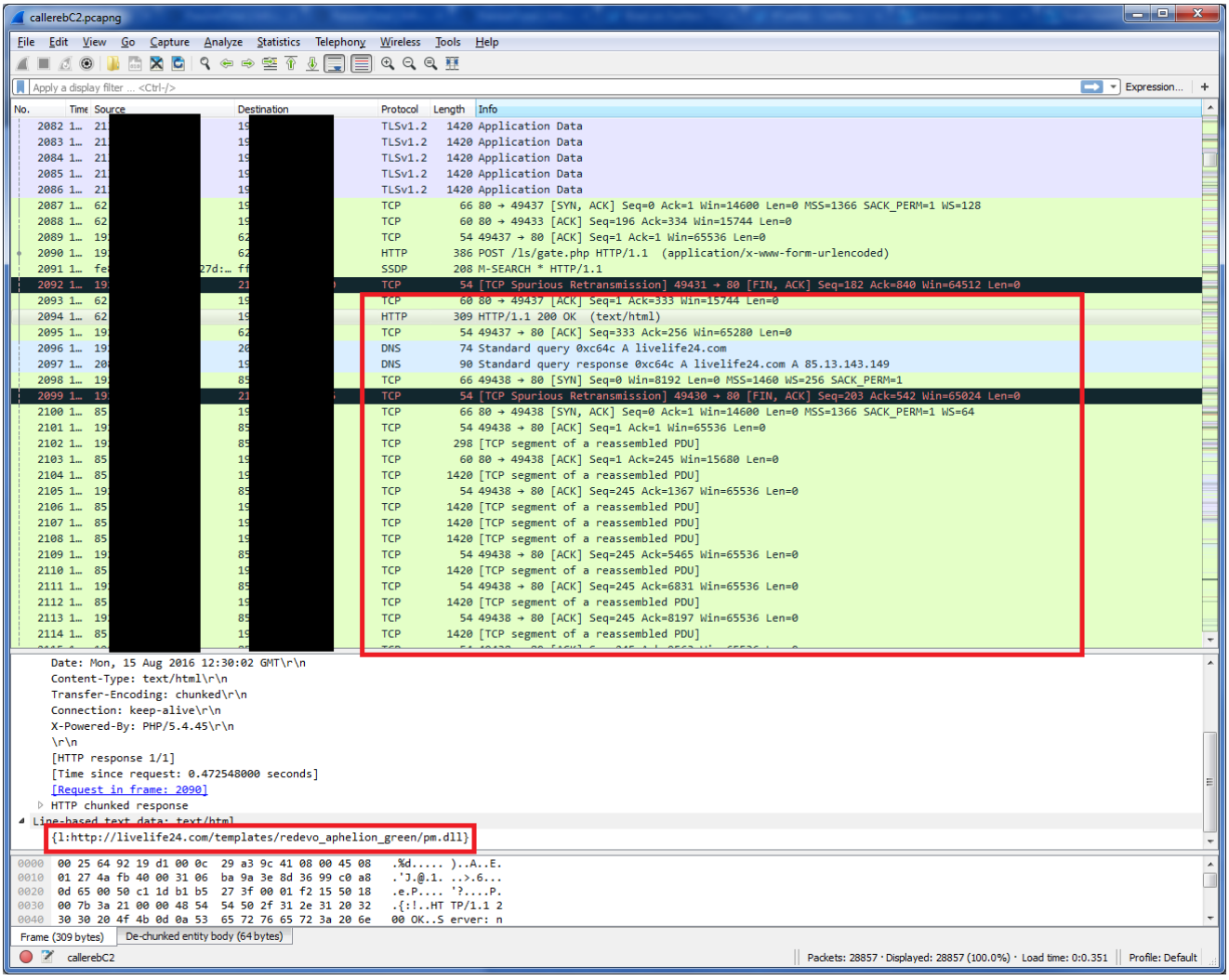
Hancitor creates svchost.exe instance to host malicious code

Proofpoint's researchers predicted that downloaders will get more complex, absorbing functionality of later stages in the infection process – our findings certainly support their assertions.

Payloads – More of the Same

After allowing the Hancitor sample to run in a controlled environment we were able to intercept it and downloaded a couple of modules. Both modules were downloaded from WordPress and Joomla! sites, possibly exploited to store the malicious content.

The first payload we observed was a Pony info-stealer Trojan (VT):



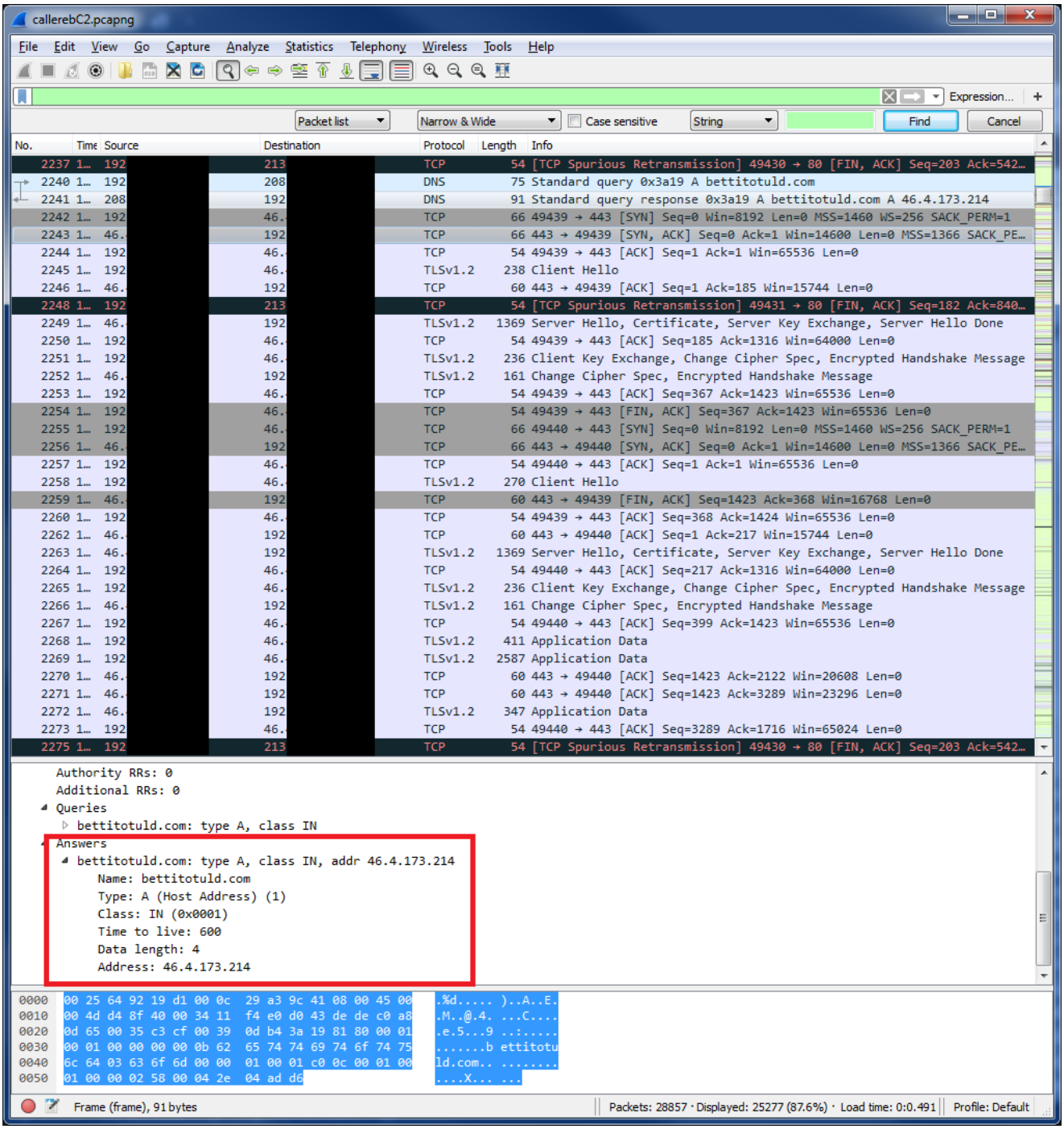
Command to download the Pony malware

After downloading it directly to Hancitor's memory it was executed in a new thread and started to monitor a vast range of collectible data:

- Email passwords (SMTP, POP3, IMAP)
- Other web protocols passwords (HTTP, FTP, NNTP)

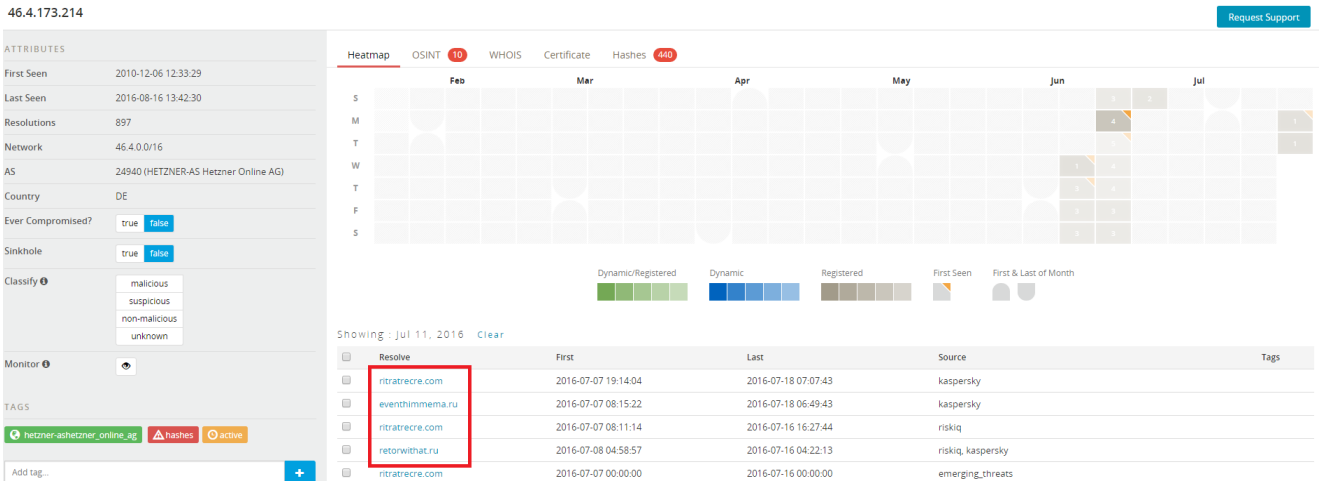
- Enumerating keys in *HKEY_CURRENT_USER\Software\Microsoft\Office\1x.0\Outlook\Profiles* leading to the users PST files
- More registry keys related to outlook accounts.

Comparing this sample to older Pony from April to early July resulted in little to no difference. Even though some of the C2 URLs were changed, we discovered that they resolve to the same IP addresses used in previous campaign. This is our sample, resolving *bettitotuld[.]com*:



Pony resolves its C2 to 46[.]4[.]173[.]214

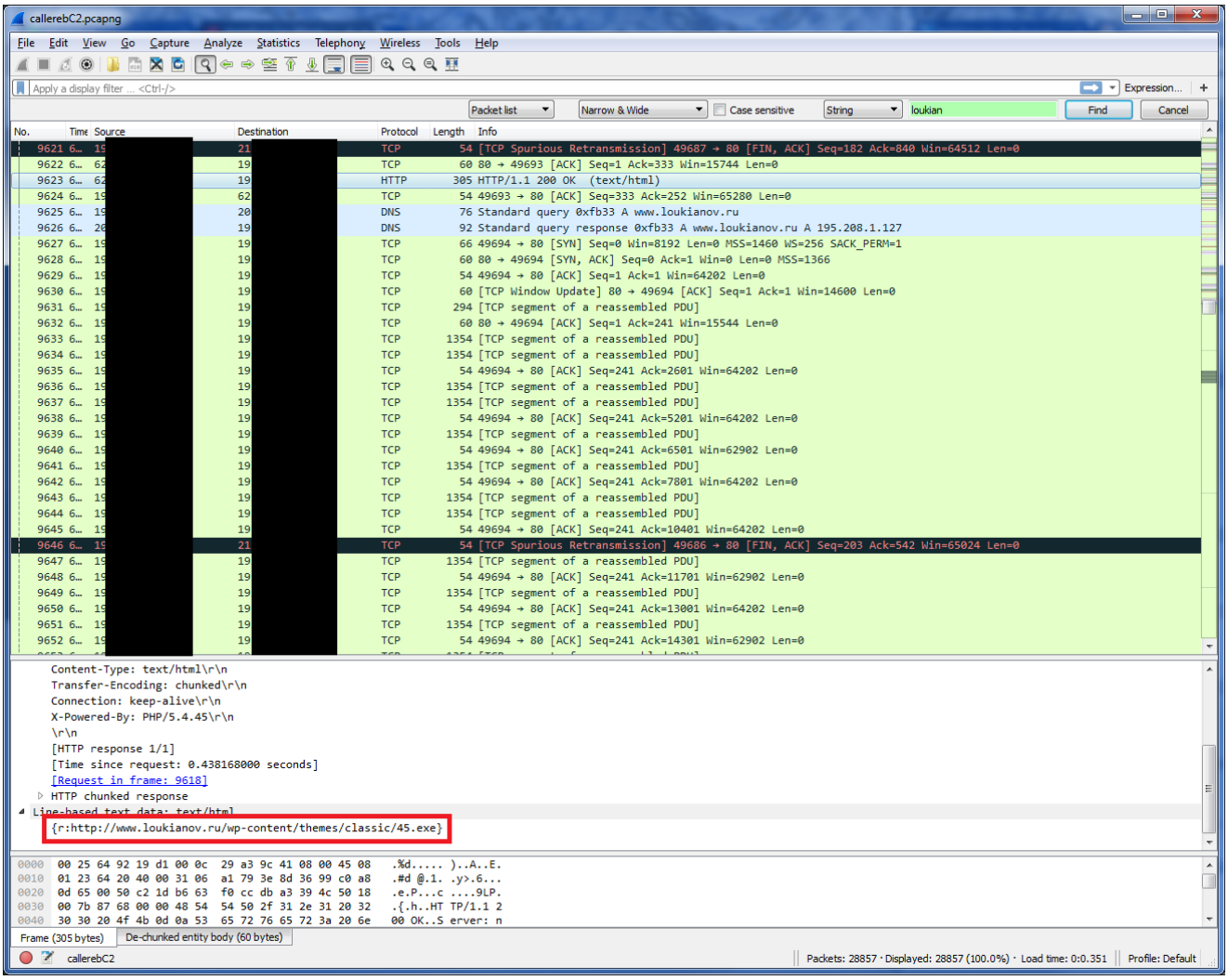
Going through Passive Total's data showed that at least four extra URLs linking this IP to previous Pony campaigns:



Same IP, different C2 URLs

In both the old and the new Pony samples the path to the gate was always the same, accessing it in the */zapoy/gate.php* path. Curious what *Zapoy* means we opened our Russian dictionary and found two possible explanations: The first one translates zapoy as the Russian term for a state of continuous drunkenness. The other meaning is slang for "start to sing".

After this short lesson in Russian slang we went back to check how Hancitor is doing and found that our sample downloaded and executed another component:



Downloading secondary payload

This file, *45.exe*, is a spam bot which was executed after being written to the disk (unlike Pony's DLL).

After a short "chat" over UDP with its C2 server the bot started to resolve the addresses of SMTP servers and connect to them over port 25:

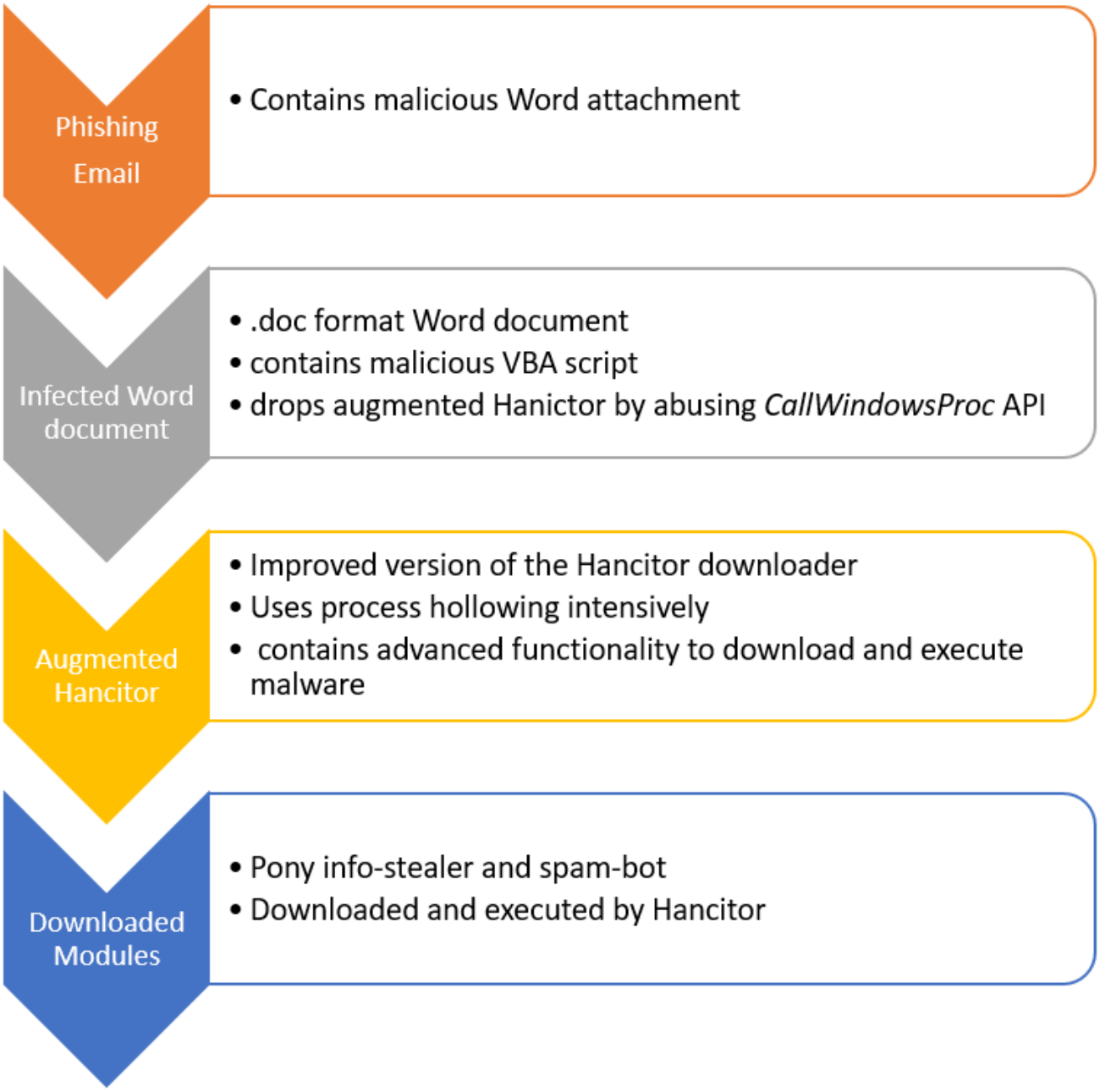
```
66 50277 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
66 50278 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
66 50279 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
81 Standard query 0xaa00 A smtp.secureserver.net
66 50280 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
66 50281 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
66 50282 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
84 Standard query 0xab00 A auscadpaging.acadian.com
66 50283 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
66 50284 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
97 Standard query response 0xaa00 A smtp.secureserver.net A 68.178.213.203
66 50285 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
105 Standard query 0xac00 A danieldefense-com.mail.protection.outlook.com
66 50286 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
100 Standard query response 0xab00 A auscadpaging.acadian.com A 199.48.249.21
66 50287 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
66 50288 → 25 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
```

The bot searches SMTP servers

It is also worth mentioning that this bot has a separate persistency mechanism than Hancitor's, installing itself as a service under the name "*s3svc*".

A Short Summary and Conclusions

The new version of Hancitor is just another phase on the evolution of downloaders from a simple “check-updates-download-execute” loop to a complex and more advanced malware. In this example we had the chance to observe the full chain from a phishing email to a Trojan:



This complex mechanism is a result of the current security products landscape – each evasive maneuver is tweaked to avoid a specific class of products. Minerva Anti-Evasion Platform, preventing any damage by this and other malware attack by exploiting malware evasive nature against it self.

IOCs

URL Addresses

Hancitor

hxxp://callereb[.]com/ls/gate[.]php

hxxp://supketwron[.]ru/ls/gate[.]php

hxxp://witjono[.]ru/ls/gate[.]php

Pony

hxxp://eventtorshendint[.]ru/zapoy/gate[.]php

hxxp://tefaverrol[.]ru/zapoy/gate[.]php

hxxp://bettitotuld[.]com/zapoy/gate[.]php

hxxp://tonslacsotont[.]ru/zapoy/gate[.]php

hxxp://hinhenharre[.]ru/zapoy/gate[.]php

hxxp://helahatun[.]com/zapoy/gate[.]php

hxxp://idmuchatbut[.]ru/zapoy/gate[.]php

hxxp://dafiutrat[.]ru/zapoy/gate[.]php

hxxp://onketorsco[.]com/zapoy/gate[.]php

hxxp://eventtorshendint[.]ru/zapoy/gate[.]php

IP Addresses

62[.]141[.]54[.]153

151[.]80[.]220[.]47

185[.]31[.]160[.]190

185[.]46[.]8[.]214

46[.]4[.]173[.]214

91[.]220[.]131[.]45

Hashes (SHA-256)

Infected Word Documents

8d37d622baf17eaa7a0b04ab1956263abcc4cd6d85fd28945aacf0dac87b47c4

fcc24a15f2b7ed06403ec192b3ed2a5258e2691b6d61b2334160fd76bbfba151

9463dc78dc7df3e751ee8c10a3fa32e315f58924eb0305f5f9eeaeae2865f9dd

21efc8907d1c4f320330da3f6a87030f1c389ac8d4fc7363d170ce9444ec81cd

554ff7c6f98afd3c6d9aaef232748481c8024feef415dcf4e153cdbed1a3994e

7edd4f271ae83b5c13b9d1927b9a64160d5ffa2eab88e9a860e50009385638a7

4b99b55479698ee6d1f6b69999c994e153672706af477c84cee6858240569783

cc07a2baf22c94959623b1a89ed88a317dbd7a131d4cdc3eadb048f32b3a2e7b

29f99f50e0aecd0e3c41c7dc1ecdfbc52fb53f734d0de99b5ff722dd07149173

926a34fbae94ab7ed7fe9a596f0507031e19044c06cbbca245efb30d926ea1e5

d59bceef11d49f47ec956b7bc9d3497ffc5259905cd6797ff9f5384f0ee55521

af3d08fb9f2e2ba73496aebb53d36dae1d812622abd598eba27c5d483129632d

ac7a5bfc346193a43e6e22663c1037ca45d89a92c8bb3cefb165c359abb402c4

c1ab4f0d1184df1be78d202e1a204fe187eb1649b1e912b48c6eef46af89c430

37a4084541df61d1380370a59694ba6c59abebf0c8183e10abe60d17bdeacedd

1b6e050c9f5fdcb04b247ef9db8fa2a6322118ed7b71c1545d39cb25a1e16131

Hancitor

fcc24a15f2b7ed06403ec192b3ed2a5258e2691b6d61b2334160fd76bbfba151

9463dc78dc7df3e751ee8c10a3fa32e315f58924eb0305f5f9eeaeae2865f9dd

21efc8907d1c4f320330da3f6a87030f1c389ac8d4fc7363d170ce9444ec81cd

554ff7c6f98afd3c6d9aaef232748481c8024feef415dcf4e153cdbed1a3994e

7edd4f271ae83b5c13b9d1927b9a64160d5ffa2eab88e9a860e50009385638a7

4b99b55479698ee6d1f6b69999c994e153672706af477c84cee6858240569783

cc07a2baf22c94959623b1a89ed88a317dbd7a131d4cdc3eadb048f32b3a2e7b

29f99f50e0aecd0e3c41c7dc1ecdfbc52fb53f734d0de99b5ff722dd07149173

926a34fbae94ab7ed7fe9a596f0507031e19044c06cbbca245efb30d926ea1e5

d59bceef11d49f47ec956b7bc9d3497ffc5259905cd6797ff9f5384f0ee55521

af3d08fb9f2e2ba73496aebb53d36dae1d812622abd598eba27c5d483129632d

ac7a5bfc346193a43e6e22663c1037ca45d89a92c8bb3cefb165c359abb402c4

c1ab4f0d1184df1be78d202e1a204fe187eb1649b1e912b48c6eef46af89c430

37a4084541df61d1380370a59694ba6c59abebf0c8183e10abe60d17bdeacedd

1b6e050c9f5fdcb04b247ef9db8fa2a6322118ed7b71c1545d39cb25a1e16131

Pony

8d60356e89c0f4d735e665bbc10c8a36589413f55efa17659c7c253d2449d54f

Spam Bot

b4e5f56345757fbea0dee5480267551c08e9d91d58960463be4928f69c89313c

99824a0be3c3922c564419e5d42dbbc0ccfbbe5f4226e74afb2ec0cada18a152t