BackDoor.Whitebird.1 — How to quickly look up a virus in the Dr.Web virus database

Published: 2020-07-14 · Archived: 2026-04-06 03:25:52 UTC

Packer: absent

Compilation date: 28.05.2018 23:14:08

SHA1 hash:

- e70a5ce00b3920d83810496eab6b0d028c5f746e

Description

A multifunctional backdoor trojan for Microsoft Windows 64-bit operating systems. Its function is to establish an encrypted connection with the C&C server and grant unauthorized access to the infected computer. It has a file manager, proxy server and remote shell capabilities. It was used in targeted attacks on state institutions in Kazakhstan and Kyrgyzstan. Similar to **BackDoor.PlugX**, this modification was used to infiltrate the network infrastructure.

Operating principle

The trojan represents a dynamic library with the MyInstall exported function. Upon infecting the targeted system, it is installed in the C:\Windows\System32\oci.dll directory.

The program launches as follows. Upon operating system boot, a Microsoft Distributed Transaction Coordinator (MSDTC) is launched. The Windows registry contains the parameters of this service, which hold the names of the loading libraries. By default, the OracleOciLib and OracleOciLibPath keys in the HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSDTC\MTxOCI branch have the values of oci.dll and %systemroot%\system32 accordingly. . When the trojan is placed in %systemroot%\system32\oci.dll, it will be automatically loaded onto the memory when the MSDTC starts.

When initialized, it creates a gfhfgh6y76734d,1111 mutex, followed by the library loading and the MyInstall exported function call.

MyInstall

The trojan can determine if the proxy server should be used and can perform a basic authentication and authorization via the NTLM protocol. When running, it logs records in the journal, saving it as c:\programdata\logos.txt.

It connects to the C&C server and exchanges the keys with it. All subsequent packets between the trojan and the server are encrypted. The algorithm based on the XOR operation with the buffer length of 28 bytes is used for

decryption. All packets are encrypted with an end-to-end offset in the buffer; but for the encryption and decryption, separate counters are used.

The following structure is used to request commands from the server and send the results:

```
 #pragma pack(push, 1)
struct st_getcmd
{
  _DWORD sig;
  _DWORD cmd;
  _DWORD res;
  _DWORD dwordc;
};
#pragma pack(pop)
```

The sig parameter always has a 0x03 value. To request the command from the server, the cmd parameter is set as 0x200, and the res and dwordc parameters are set to zero. If the server does not send any data within 44 seconds, the trojan sends a packet containing the cmd parameter with the 0x00 value. This process repeats until any response is received from the server.

Commands list

The commands the trojan can execute, as well as its response to them, are shown below:

- 0x00 — lack of the reply, awaiting the next command;
- 0x01 (collecting information about the bot) — replies with the cmd_botinfo structure:

```
 #pragma pack(push, 1)
struct cmd_botinfo_int
{
  _DWORD sig; // 0x03
  _DWORD OSMajorVersion;
  _DWORD OSMinorVersion;
  _DWORD OSPlatformId;
  _DWORD userpriv;
  _DWORD botip;
  _QWORD MemTotalPhys;
  _BYTE macaddr[6];
  wchar_t szCSDVersion[128];
  wchar_t hostname[64];
  wchar_t username[64];
  char connect_string[256];
};

struct cmd_botinfo
{
```

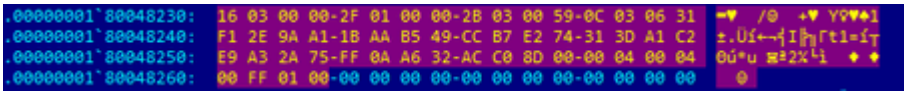
```
  _BYTE sig; // 0x03
  _WORD len; // 0x3AC
  _WORD cmdid;
  _BYTE gap[10];
  cmd_botinfo_int info;
};
#pragma pack(pop)
```

- 0x02 (remote shell launch) — replies with the packet, similar to the one received from the server;
- 0x03 (advanced file system manager launch) — replies with the packet, similar to the one received from the server;
- 0x05 (remote shell v2 launch) — replies with the packet, similar to the one received from the server;
- 0x06 (proxy manager launch) — replies with the packet, similar to the one received from the server;
- 0x100 (the ping command) — replies with cmd=0x00;
- 0x400 (the command to reconnect to the server) — replies with cmd=0x300;
- 0x600 (dummy command) — replies with cmd=0x600; res=0xffffffff;
- 0x700 (launch of the command through ShellExecute) — replies with cmd=0x700; if failed, replies with res=0xffffffff.

Exchanging keys

The process of exchanging keys with the C&C server is as follows:

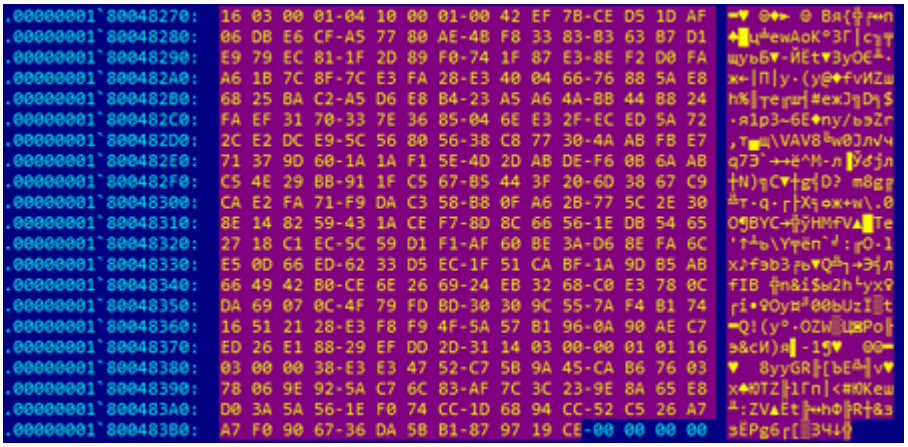
Using random values, the trojan initializes the buffer with the size of 28 bytes. Next, it takes the data array of the 58 bytes size, which is embedded into its body.



It encrypts bytes from 15 to 43, based on the XOR operation algorithm, using randomly generated bytes, and sends the resulting buffer to the server. In response, it should receive 5 bytes, where 0x16 is a 0 byte and the htons function results from WORD, starting with the third byte, which is the size of the next packet, and shouldn't exceed 0x3FF9 bytes.

After that, it receives the next packet, whose data is used in the next exchange.

Next, the trojan uses the second encoded buffer with the size of 332 bytes.



The trojan encrypts the bytes, starting from 9 to 265 and from 304 to 332, with the algorithm based on the XOR operation, using randomly generated bytes. 28 bytes, starting from 276 bytes, is replaced with the data generated upon the first buffer initialization. There must be a response of 5 bytes, where the 0 byte is 0x14, and the htons function results from WORD, starting with the 3rd byte, which is the size of next packet, and should not exceed 0x3FF9 bytes.

After that, it receives the next packet, whose data is not used in further exchange.

Next, the trojan receives 5 bytes from the C&C server, where 0x16 is the 0 byte, and the htons function results from WORD, starting with the 3rd byte, which is the size of the next packet, and should not exceed 0x38 bytes.

It receives the next packet from the C&C server and sends 0x38 bytes into the encryption key initialization function:

```
 __int64 __fastcall CCrypt::GenKeys(ccrypt *this, _BYTE *ext_key)
{
  __int64 result; // rax
  int i; // [rsp+0h] [rbp-18h]
  for ( i = 0; i < 28; ++i )
  {
    this->key[i] ^= ext_key[i];
    this->key[i] ^= ~(_BYTE)i;
    if ( !this->key[i] )
      this->key[i] = ~(_BYTE)i;
    result = (unsigned int)(i + 1);
  }
  return result;
}
```

Remote Shell Function

The trojan copies %WINDIR%\System32\cmd.exe into %WINDIR%\alg.exe. It then initializes a new connection to the C&C server and sends the following packet:

```
 #pragma pack(push,1)
struct cmd_remoteshell
{
  _WORD sig; // 0x03
  _WORD len;
  _WORD cmd; // 0x02
  _BYTE gap[10];
  _BYTE macaddr[6];
};
#pragma pack(pop)
```

Next, it launches a scanned alg.exe with the pipes input/output redirection. If the launch fails, it runs a cmd.exe instead of the alg.exe. If there is data in the output function pipe, the trojan sends the data to the server in the following packet:

```
 #pragma pack(push,1)
struct cmd_remoteshell_out
{
  _WORD sig; // 0x03
  _WORD len;
  _WORD cmd; // 0x202
  _BYTE gap[10];
  wchar_t buffer[];
};
#pragma pack(pop)
```

Herewith, the trojan periodically checks for data from the C&C server and parses the incoming command when the data has been received.

List of Remote Shell Commands

Command	Description	Argument	Response
0x100	keep-alive mode	-	cmd = 0x00
0x102	executes the command in the Remote Shell	a command	-
0x103	launches the file manager (writing into the end of existing file)	a path to the file, the final size of the file	cmd value is identical to the value in the packet received from the server; res = -1 if failed; res = 0 if succeed.
0x203	launches the file manager (reading from the file)	a path to the executable file, an offset in the file	

Command	Description	Argument	Response
0x703	launches an application	a path to the executable file and arguments	res = -1 if failed; res = 0 if succeed.
the remaining variants	default behavior	-	cmd value is identical to the value of the packet received from the server; res = 1.

Remote Shell v2

The trojan copies %WINDIR%\System32\cmd.exe into the %WINDIR%\alg.exe. It then initializes a new connection to the C&C server and sends the following packet:

```
 #pragma pack(push,1)
struct cmd_remoteshell
{
  _WORD sig; // 0x03
  _WORD len;
  _WORD cmd; // 0x02
  _BYTE gap[10];
  _BYTE macaddr[6];
};
#pragma pack(pop)
```

Next, it launches a copied alg.exe; if launch has failed, it runs a cmd.exe instead of the alg.exe. Input/output to the launched process is implemented via the trojan process joining to the console of the launched alg.exe/cmd.exe process, using the WINAPI AttachConsole.

The rest of the operation routine is similar to the one in the Reverse Shell handler.

File manager

The trojan initializes a new connection to the C&C server and sends the following packet:

```
 #pragma pack(push,1)
struct cmd_fileop
{
  _WORD sig; // 0x03
  _WORD len;
  _WORD cmd;
  _WORD gap;
  _DWORD res;
```

```
  _DWORD filesize;
  _BYTE macaddr[6];
};
#pragma pack(pop)
```

The cmd value is set to the same value in the server packet. Next, the trojan receives commands from the server.

- 0x103:

  Checks for the file availability. If it does not exist, it sends the packet with the res = 0xB7 value; Tries to open the file in append mode. If failed, it sends the packet with the res = 0x52 value;

  Receives the file size and sets filesize filed to the corresponding value in the subsequent packets;

  Receives packets in a cycle with the cmd = 0x303 packet value and writes the data into the file until the file size is larger or equal to the one the server indicated in the first packet.

- 0x203:

  Tries to open the file in reading mode. If failed, it sends the packet with the res = 0x02 value;

  Receives the file size and sends it to the server in the packet;

  In a cycle, it reads the file, starting from the offset, which is indicated in filesize the first packet received from the server, and sends the data in the packet with the cmd = 0x303 value to the server until the file hasn’t been read to its end.

- 0x403:

  If the C&C server sends the path as an argument, the trojan lists the files and folders available in this path (not recursively) and sends the collected information with the cmd = 0x403 value to the server;

  If the C&C server does not specify the argument or if the first symbol of the argument is '/' or '\\', the trojan lists every storage device and collects the data, including the disk type, its size and free space available, and then sends this data to the server in the packet with the cmd = 0x403 value.

- 0x503:

  Moves a file (the initial and final paths are specified by the C&C server). In response, it sends the packet with the cmd = 0x503 and res = 0 values if succeeded; otherwise, it sends the packet with the res = -1 value.

- 0x603:

  Deletes the file located in the path, specified by the server. In response, it sends the packet with the cmd = 0x603 and res = 0 values if succeeded; otherwise, it sends the packet with the res = -1 value.

- 0x703:

Launches an application specified by the server by using specific arguments. In response, it sends the packet with the cmd = 0x703 and res = 0 values if succeeded; otherwise, it sends the packet with the res = -1 value.

Proxy manager

The trojan initiates a new connection to the server and sends the following packet to it:

```
 #pragma pack(push,1)
struct cmd_proxy
{
  _WORD sig; // 0x03
  _WORD len;
  _WORD cmd; // 0x06
  _BYTE gap[10];
  _BYTE macaddr[6];
};
#pragma pack(pop)
```

Next, it receives the commands from the server.

- 0x106:

  Opens one of the available ports;

  Sends a packet with the cmd = 0x506 value to the server;

  Connects to the targeted server using the IP and port, specified by the C&C server;

  Waits for the incoming connection to its port. Upon receiving the data, it sends it to the server it is connected to;

  If the trojan receives the data from the targeted server, it sends it to the C&C server in the packet with the cmd = 0x116 value;

  Returns to waiting for the incoming connection to its port. Upon receiving the data, it sends it to the server it is connected to.

- 0x116:

  If there is an incoming connection to a previously opened port, the trojan sends the raw data to the client without using the encryption standard to the trojan.

- 0x126:

  Stops the proxy and closes all opened connections.

- 0x206:

Sends the packet with the cmd = 0x506 value to the C&C server;

Opens a port specified by the server;

Waits for the incoming connection to the specified port;

Connects to the targeted server specified by the C&C server;

Forwards the traffic from the local port to the remote server and backwards as raw data, not using the encryption, standard to the trojan.

- 0x306:

  Receives two ports as an argument;

  Sends the packet with the cmd = 0x506 value to the C&C server;

  Opens first port (master port) and waits for the connection;

  Opens the second port (client port) and waits for the connection;

  Opens a random port and sends its number to the target, which is currently connected to the master port. Next, it waits for the incoming connection on the specified port;

  Forwards the traffic between the clients, which connected to the master port and random port.

- 0x406:

  Receives two pairs of IP:port as an argument;

  Connects to the first server and receives 2 bytes from it, which are the port number;

  Connects to the same server through the received port;

  Connects to the second server, specified in the incoming arguments;

  Forward the traffic between previously established connections.

- 0x606:

  Stops proxy server operation.
