# BackDoor.Crane.1

**Added to Dr.Web virus database:** 2016-10-21

**Virus description was added:** 2016-11-18

http://vms.drweb.com/virus/?_is=1&i=8986771

**SHA1:**

- afecbbc9d6dbc326fa448674b7b252c05e3c0c9b - BackDoor.Crane.1
- 33817963d25d8bf45bbaf8c09e6e82a26532b4e3 - CmdModule.dll
- 61e7f7a02d8b63af4b3738ca6086a442a4a72359 - DownloadModule.dll
- 5596a021c1a7532c876c13bf0645b863b322cf84 - FilelistingModule.dll
- 96ca14351bd7e34536b0cd945c3d952bfb0a5f6d - ScreenshotModule.dll
- 3aa2c53361d79172c0f6e8a57fad7f9f14667bf2 - UploadModule.dll
- 6a1d24439c15aa559b34e411f2f52d8fb564dba1 - UploadToAdminModule.dll

A modular backdoor used by cybercriminals to carry out targeted attacks on largest Russian companies producing portal and lifting cranes as well as auxiliary equipment. It is written in C++. The last compilation of the examined sample is Thu Apr 21 10:56:19 2016. The path to debug information:

```
C:\Users\User\Desktop\14\bin\Bot.pdb
```

Once launched, the Trojan creates a separate thread to perform its malicious activity. It then checks the configuration file connfig.jsonin the catalog "%ALLUSERSPROFILE%\\yandex_service". If the file is found, the Trojan gets necessary data from it. If not, the Trojan creates this file with a default content:

```
{
```

```
    "frequency" : 60,
```

```
    "id" : "********",
    "proxy" :
    {
        "host" : "",
        "pass" : "",
        "port" : 0,
        "protocol" : 0,
        "user" : ""
    },
    "script_name" : "***.php",
    "servers" :
    [
        {
            "host" : "*****.**",
            "port" : 80,
            "useSSL" : false
        }
    ]
```

```
}
```

After a name of a folder is determined, the Trojan searches its own modules scanning files by their masks:

```
 < dir>\\modules\\*Module.dll
```

Found modules are then loaded to the memory using LoadLibrary. Meanwhile, the Trojan checks that the library has the exports GetModuleName, GetModuleCommand, CreateModule, DeleteModule, and RunCommand.

Once the modules are loaded the Trojan periodically requests to the C&C server for further instructions. First, it registers on the server and sends information about an infected computer: its name, a user name, an IP address, a list of the running processes, and a randomly generated bot identifier. It should be noted that, while sharing data with the server, the Trojan uses the string “RSDN HTTP Reader” as a value of the parameter User-Agent.

```
{
    "action":"auth",
    "controller":"api",
    "data":"
        {
            \"computer_name\":\"*****\",
            \"frequency\":60,
            \"id\":\"*****\",
            \"network_interfaces\":[\"A.B.C.D\"],
```

```
\"processes\":[\"System\",\"smss.exe\",\"csrss.exe\",\"
wininit.exe\",\"csrss.exe\",\"winlogon.exe\",\"services
.exe\",\"lsass.exe\",\"lsm.exe\",\"svchost.exe\",\"svch
ost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe
\",\"svchost.exe\",\"svchost.exe\",\"explorer.exe\",\"s
poolsv.exe\",\"taskhost.exe\",\"svchost.exe\",
```

```
            \"proxy\":0,
```

```
            \"script_name\":\"***.php\",
```

```
\"servers\":[{\"host\":\"***\",\"port\":80,\"userSSL\":
false}],\"user_name\":\"user\"
```

```
        }\n"
```

```
}
```

As a response to this request, the server sends the confirmation:

```
{"status":"SUCCESS"}
```

The Trojan then awaits for instructions after sending GET requests. The server's replies are in the JSON format:

```
{"status":"SUCCESS","data":null}
```

This example shoes that the field “data” does not contain information, so that the bot does not need to execute any commands. When sending a command, the field “data” has the following structure:

```
{
    "id": <bot_id>,
    "command": {
        "command": <command_name>,
        "cmd": {
            "wait": true
        }
    }
}
```

where <bot_id> corresponds to the bot’s identifier, <command_name> is name of a command or a module for which the command is intended. The Trojan can execute only one command—“updateConf”—that updates the configuration file. Other commands are executed by the modules.

## Modules

All the modules are named *Module.dll, located in the directory “<dir>\\modules\\”, and has the exports GetModuleName, GetModuleCommand, CreateModule, DeleteModule, and RunCommand.

### CreateModule

It creates the structure st_module and returns the pointer on it:

```
struct st_module
{
  char * name;
  char * command;
};
```

**GetModuleName**

It returns the field “name” of the structure st_module.

**BackDoor.Crane.1**

**Added to Dr.Web virus database: 2016-10-21**

**Virus description was added: 2016-11-18**

**SHA1:**

**afecbbc9d6dbc326fa448674b7b252c05e3c0c9b - BackDoor.Crane.1**

**33817963d25d8bf45bbaf8c09e6e82a26532b4e3 - CmdModule.dll**

**61e7f7a02d8b63af4b3738ca6086a442a4a72359 - DownloadModule.dll**

**5596a021c1a7532c876c13bf0645b863b322cf84 - FilelistingModule.dll**

**96ca14351bd7e34536b0cd945c3d952bfb0a5f6d - ScreenshotModule.dll**

**3aa2c53361d79172c0f6e8a57fad7f9f14667bf2 - UploadModule.dll**

**6a1d24439c15aa559b34e411f2f52d8fb564dba1 - UploadToAdminModule.dll**

**A modular backdoor used by cybercriminals to carry out targeted attacks on largest Russian companies producing portal and lifting cranes as well as auxiliary equipment. It is written in C++. The last compilation of the examined sample is Thu Apr 21 10:56:19 2016. The path to debug information:**

**C:\Users\User\Desktop\14\bin\Bot.pdb**

**Once launched, the Trojan creates a separate thread to perform its malicious activity. It then checks the configuration file connfig.jsonin the catalog "%ALLUSERSPROFILE%\\yandex_service". If the file is found, the Trojan gets necessary data from it. If not, the Trojan creates this file with a default content:**

```
{
    "frequency" : 60,
    "id" : "*******",
    "proxy" :
    {
        "host" : "",
        "pass" : "",
        "port" : 0,
        "protocol" : 0,
        "user" : ""
    },
    "script_name" : "***.php",
    "servers" :
    [
        {
            "host" : "*****.**",
            "port" : 80,
            "useSSL" : false
        }
    ]
}
```

After a name of a folder is determined, the Trojan searches its own modules scanning files by their masks:

< dir>\\modules\\*Module.dll

Found modules are then loaded to the memory using LoadLibrary. Meanwhile, the Trojan checks that the library has the exports GetModuleName, GetModuleCommand, CreateModule, DeleteModule, and RunCommand.

Once the modules are loaded the Trojan periodically requests to the C&C server for further instructions. First, it registers on the server and sends information about an infected computer: its name, a user name, an IP address, a list of the running processes, and a randomly generated bot identifier. It should be noted that, while sharing data with the server, the Trojan uses the string “RSDN HTTP Reader” as a value of the parameter User-Agent.

```
{
    "action":"auth",
    "controller":"api",
    "data":"
        {
            \"computer_name\":\"*****\",
            \"frequency\":60,
            \"id\":\"*****\",
            \"network_interfaces\":[\"A.B.C.D\"],

\"processes\":[\"System\",\"smss.exe\",\"csrss.exe\",\"wininit.exe\",\"csrss.exe\
",\"winlogon.exe\",\"services.exe\",\"lsass.exe\",\"lsm.exe\",\"svchost.exe\",\"sv
chost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"svc
host.exe\",\"explorer.exe\",\"spoolsv.exe\",\"taskhost.exe\",\"svchost.exe\",
            \"proxy\":0,
            \"script_name\":\"***.php\",
\"servers\":[{\"host\":\"***\",\"port\":80,\"userSSL\":false}],\"user_name\":\"
user\"

        }\n"

}
```

As a response to this request, the server sends the confirmation:

```
{"status":"SUCCESS"}
```

The Trojan then awaits for instructions after sending GET requests. The server's replies are in the JSON format:

```
{"status":"SUCCESS","data":null}
```

This example shoes that the field "data" does not contain information, so that the bot does not need to execute any commands. When sending a command, the field "data" has the following structure:

```
{

    "id": <bot_id>,

    "command": {

        "command": <command_name>,

        "cmd": {

            "wait": true

        }

    }

}
```

where <bot_id> corresponds to the bot's identifier, <command_name> is name of a command or a module for which the command is intended. The Trojan can execute only one command—"updateConf"—that updates the configuration file. Other commands are executed by the modules.

Modules

All the modules are named *Module.dll, located in the directory "<dir>\\modules\\", and has the exports GetModuleName, GetModuleCommand, CreateModule, DeleteModule, and RunCommand.

CreateModule

It creates the structure st_module and returns the pointer on it:

```
struct st_module

{

   char * name;

   char * command;

};
```

GetModuleName

It returns the field "name" of the structure st_module.

GetModuleCommand

It returns the field "cmd" of the structure st_module.

DeleteModule

Clears the memory used for creation of the structure st_module.

RunCommand

Executes the specified command.

CmdModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\CmdModule.pdb

name: "CmdModule"

command: "cmd"

IT executes the command using the command interpreter cmd.

DownloadModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\DownloadModule.pdb

name: "DownloadModule"

command: "download"

It downloads a file from a specified link and saves it to a certain folder on a computer.

FilelistingModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\FilelistingModule.pdb

name: "FilelistingModule"

command: "filelisting"

It generates a list of folder content and sends to the C&C server.

ScreenshotModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\ScreenshotModule.pdb

name: "ScreenshotModule"

command: "screenshot"

It takes a screenshot and sends it to the C&C server.

UploadModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\UploadModule.pdb

name: "UploadModule"

command: "upload"

It loads a file to a specified server over the FTP protocol.

UploadToAdminModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\UploadToAdminModule.pdb

**name: "UploadToAdminModule"**

**command: "uploadtoadmin"**

**It loads a file to a specified server over the HTTP protocol.**

**News about the Trojan**

**GetModuleCommand**

It returns the field "cmd" of the structure st_module.

**BackDoor.Crane.1**

**Added to Dr.Web virus database: 2016-10-21**

**Virus description was added: 2016-11-18**

**SHA1:**

**afecbbc9d6dbc326fa448674b7b252c05e3c0c9b - BackDoor.Crane.1**

**33817963d25d8bf45bbaf8c09e6e82a26532b4e3 - CmdModule.dll**

**61e7f7a02d8b63af4b3738ca6086a442a4a72359 - DownloadModule.dll**

**5596a021c1a7532c876c13bf0645b863b322cf84 - FilelistingModule.dll**

**96ca14351bd7e34536b0cd945c3d952bfb0a5f6d - ScreenshotModule.dll**

**3aa2c53361d79172c0f6e8a57fad7f9f14667bf2 - UploadModule.dll**

**6a1d24439c15aa559b34e411f2f52d8fb564dba1 - UploadToAdminModule.dll**

**A modular backdoor used by cybercriminals to carry out targeted attacks on largest Russian companies producing portal and lifting cranes as well as auxiliary equipment. It is written in C++. The last compilation of the examined sample is Thu Apr 21 10:56:19 2016. The path to debug information:**

**C:\Users\User\Desktop\14\bin\Bot.pdb**

**Once launched, the Trojan creates a separate thread to perform its malicious activity. It then checks the configuration file connfig.jsonin the catalog "%ALLUSERSPROFILE%\\yandex_service". If the file is found, the Trojan gets necessary data from it. If not, the Trojan creates this file with a default content:**

```
{
    "frequency" : 60,
    "id" : "*******",
    "proxy" :
    {
        "host" : "",
        "pass" : "",
        "port" : 0,
        "protocol" : 0,
        "user" : ""
    },
    "script_name" : "***.php",
    "servers" :
    [
        {
            "host" : "*****.**",
            "port" : 80,
            "useSSL" : false
        }
    ]
}
```

After a name of a folder is determined, the Trojan searches its own modules scanning files by their masks:

```
 < dir>\\modules\\*Module.dll
```

Found modules are then loaded to the memory using LoadLibrary. Meanwhile, the Trojan checks that the library has the exports GetModuleName, GetModuleCommand, CreateModule, DeleteModule, and RunCommand.

Once the modules are loaded the Trojan periodically requests to the C&C server for further instructions. First, it registers on the server and sends information about an infected computer: its name, a user name, an IP address, a list of the running processes, and a randomly generated bot identifier. It should be noted that, while sharing data with the server, the Trojan uses the string “RSDN HTTP Reader” as a value of the parameter User-Agent.

```
{
    "action":"auth",
    "controller":"api",
    "data":"
        {
            \"computer_name\":\"*****\",
            \"frequency\":60,
            \"id\":\"*****\",
            \"network_interfaces\":[\"A.B.C.D\"],
\"processes\":[\"System\",\"smss.exe\",\"csrss.exe\",\"wininit.exe\",\"csrss.exe\",\"winlogon.exe\",\"services.exe\",\"lsass.exe\",\"lsm.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"svchost.exe\",\"explorer.exe\",\"spoolsv.exe\",\"taskhost.exe\",\"svchost.exe\",
            \"proxy\":0,
            \"script_name\":\"***.php\",
\"servers\":[{\"host\":\"***\",\"port\":80,\"userSSL\":false}],\"user_name\":\"user\"
        }\n"
}
```

As a response to this request, the server sends the confirmation:

```
{"status":"SUCCESS"}
```

The Trojan then awaits for instructions after sending GET requests. The server's replies are in the JSON format:

```
{"status":"SUCCESS","data":null}
```

This example shoes that the field "data" does not contain information, so that the bot does not need to execute any commands. When sending a command, the field "data" has the following structure:

```
{
    "id": <bot_id>,
    "command": {
        "command": <command_name>,
        "cmd": {
            "wait": true
        }
    }
}
```

where <bot_id> corresponds to the bot's identifier, <command_name> is name of a command or a module for which the command is intended. The Trojan can execute only one command—"updateConf"—that updates the configuration file. Other commands are executed by the modules.

Modules

All the modules are named *Module.dll, located in the directory "<dir>\\modules\\", and has the exports GetModuleName, GetModuleCommand, CreateModule, DeleteModule, and RunCommand.

CreateModule

It creates the structure st_module and returns the pointer on it:

```
struct st_module

{

  char * name;

  char * command;

};
```

GetModuleName

It returns the field “name” of the structure st_module.

GetModuleCommand

It returns the field “cmd” of the structure st_module.

DeleteModule

Clears the memory used for creation of the structure st_module.

RunCommand

Executes the specified command.

CmdModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\CmdModule.pdb

name: "CmdModule"

command: "cmd"

IT executes the command using the command interpreter cmd.

DownloadModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\DownloadModule.pdb

name: "DownloadModule"

command: "download"

It downloads a file from a specified link and saves it to a certain folder on a computer.

FilelistingModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\FilelistingModule.pdb

name: "FilelistingModule"

command: "filelisting"

It generates a list of folder content and sends to the C&C server.

ScreenshotModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\ScreenshotModule.pdb

name: "ScreenshotModule"

command: "screenshot"

It takes a screenshot and sends it to the C&C server.

UploadModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\UploadModule.pdb

name: "UploadModule"

command: "upload"

It loads a file to a specified server over the FTP protocol.

UploadToAdminModule

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\UploadToAdminModule.pdb

name: "UploadToAdminModule"

command: "uploadtoadmin"

It loads a file to a specified server over the HTTP protocol.

News about the Trojan

DeleteModule

Clears the memory used for creation of the structure st_module.

**RunCommand**

Executes the specified command.

**CmdModule**

The path to debug characters: C:\Users\User\Desktop\14\bin\modules\CmdModule.pdb

name: "CmdModule"
command: "cmd"

IT executes the command using the command interpreter cmd.

**DownloadModule**

The path to debug characters: C:\Users\User\Desktop\14\bin\modules\DownloadModule.pdb

name: "DownloadModule"
command: "download"

It downloads a file from a specified link and saves it to a certain folder on a computer.

**FilelistingModule**

The path to debug characters: C:\Users\User\Desktop\14\bin\modules\FilelistingModule.pdb

name: "FilelistingModule"
command: "filelisting"

It generates a list of folder content and sends to the C&C server.

**ScreenshotModule**

The path to debug characters: C:\Users\User\Desktop\14\bin\modules\ScreenshotModule.pdb

name: "ScreenshotModule"
command: "screenshot"

It takes a screenshot and sends it to the C&C server.

**UploadModule**

The path to debug characters: C:\Users\User\Desktop\14\bin\modules\UploadModule.pdb

name: "UploadModule"
command: "upload"

It loads a file to a specified server over the FTP protocol.

**UploadToAdminModule**

The path to debug characters:
C:\Users\User\Desktop\14\bin\modules\UploadToAdminModule.pdb

name: "UploadToAdminModule"
command: "uploadtoadmin"

It loads a file to a specified server over the HTTP protocol.

News about the Trojan

http://vms.drweb.com/virus/?_is=1&i=8986771