REMCOS: A New RAT In The Wild

By Floser Bacurio and Joie Salvio

Published: 2017-02-14 · Archived: 2026-04-06 00:44:28 UTC

Remcos is another RAT (Remote Administration Tool) that was first discovered being sold in hacking forums in the second half of 2016. Since then, it has been updated with more features, and just recently, we've seen its payload being distributed in the wild for the first time.

This article demonstrates how this commercialized RAT is being used in an attack, and what its latest version (v1.7.3) is capable of doing. Remcos is currently being sold from $58 to $389, depending on the license period and the maximum number of masters or clients needed.

Macro Executes Malware with High System Privilege

We discovered that the Remcos RAT is being distributed through malicious Microsoft Office documents going by the filenames of *Quotation.xls* or *Quotation.doc*, which are most probably attached to SPAM emails. The structure and behavior of these documents are very similar to the ones that we documented in our previous article, which details a malicious document macro designed to bypass Microsoft Windows' UAC security and execute malware with high privilege.

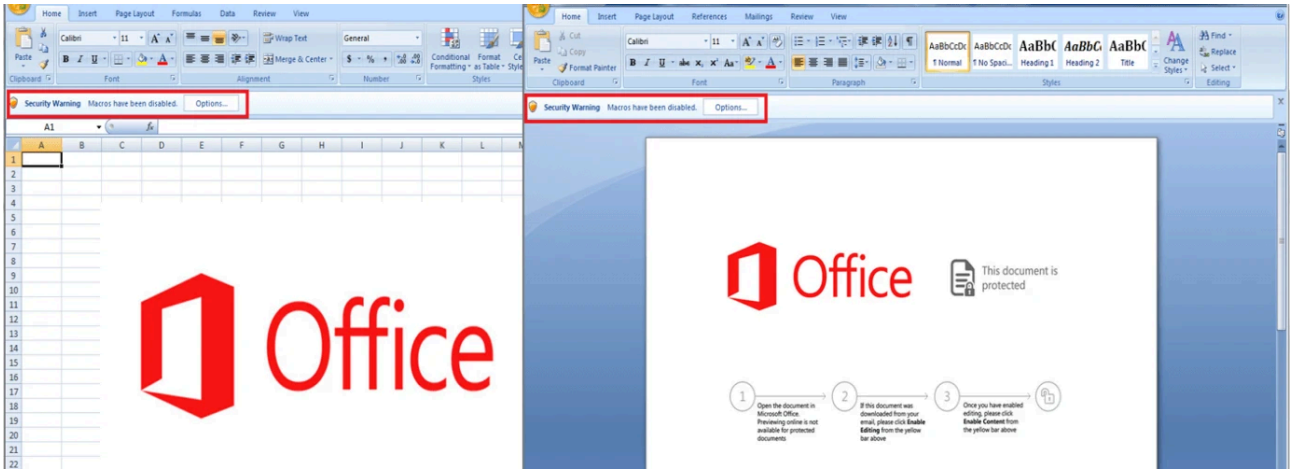

Figure 1: Malicious MS Office documents

The affected documents contain an obfuscated macro that executes a shell command that downloads and runs the malware. Its obfuscation is simply achieved by adding garbage characters to the actual string.

To execute the downloaded malware with high system privilege, it utilizes an already known UAC-bypass technique. It attempts to execute it under Microsoft's Event Viewer (eventvwr.exe) by hijacking a registry (*HKCU\Software\Classes\mscfile\shell\open\command*) that it queries to find the path of the Microsoft Management Console (mmc.exe). The Event Viewer simply executes whatever is in that path. Since the macro's shell command replaces the value from that registry entry to the malware's location, the malware is executed instead of the legitimate *mmc.exe.*

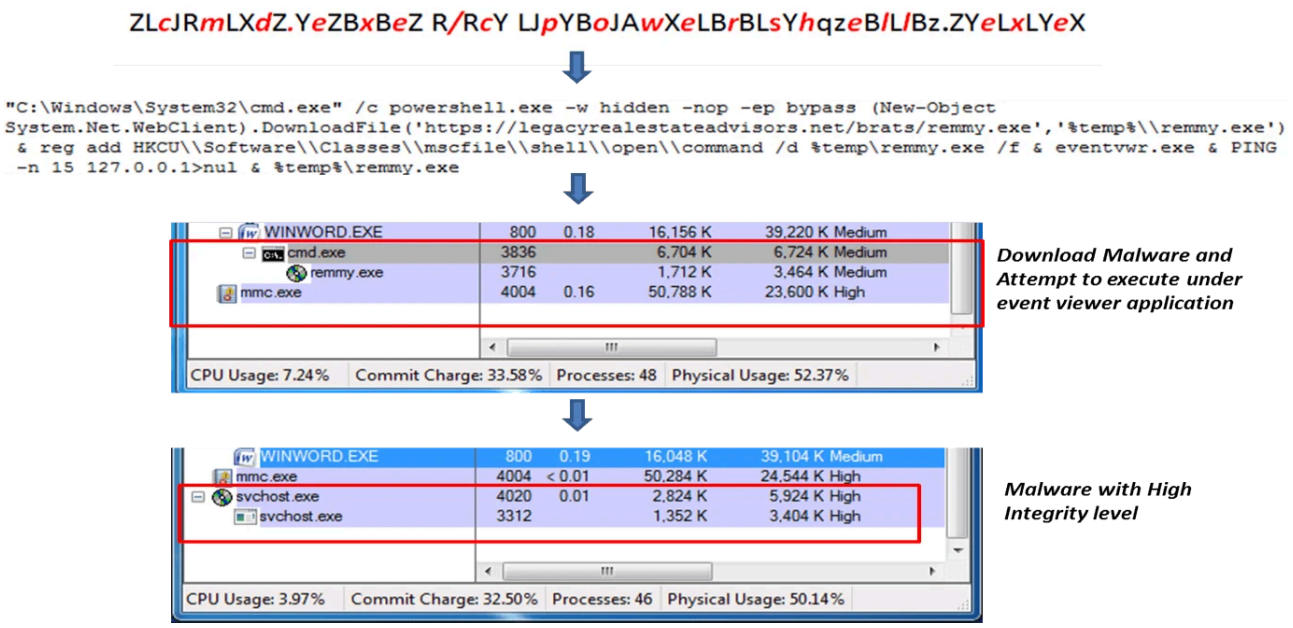

Figure 2: Execution of the malware from macro

In figure 2 we can see that when the command shell executed the downloaded malware, the integrity level was unexpectedly only set to "Medium." At this point, the UAC bypass should have worked and the malware should have been executed with "High" integrity. So we took a closer look at the shell command and found erroneous slashes ("\") in the registry path that caused the unsuccessful replacement of the registry value data. It was first thought that the technique worked, since the malware was executed with a "High" integrity level in the end. However, it was not executed under the Event Viewer. Since that attempt did not work, and yet the malware was still executed with "High" integrity level, we suspected that the malware binary itself has its own UAC-bypass technique, which was proven to be the case, as we demonstrate in the later part of this article.

Multi-packed Payload Binary

Remcos only includes UPX and MPRESS1 packers to compress and obfuscate its server component. In this sample, however, the attacker went further by adding another layer of custom packer on top of MPRESS1.

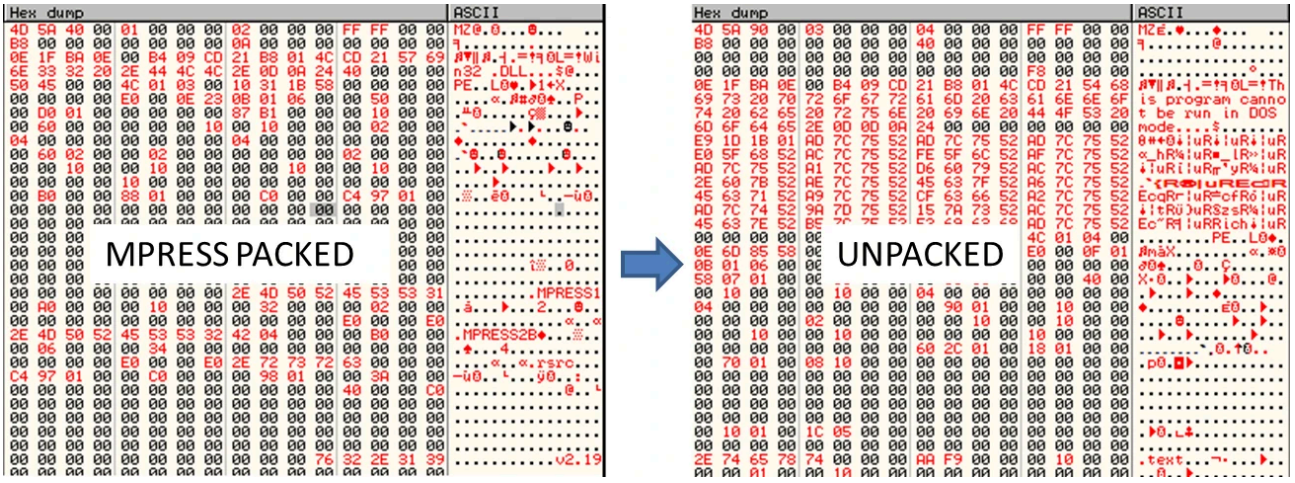

Figure 3: Hex dumps of the packed and unpacked server component

Obfuscation of the malware practically ended after the two packers. As seen in the screenshots below, the strings from the unpacked binary reveals that it's the server component built from the latest Remcos v1.7.3 Pro. According to their website, *Breaking-Security[.]Net,* this version was just released in Jan. 23, 2017.

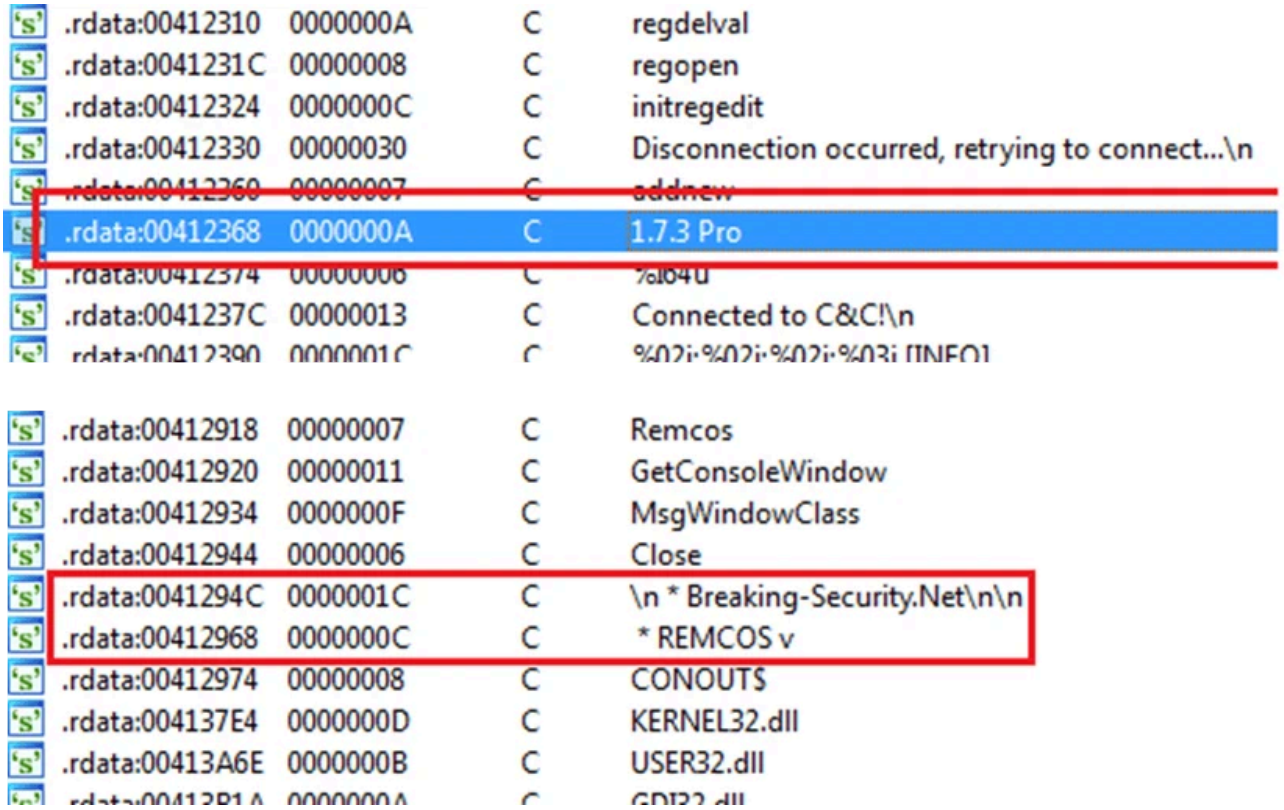

Figure 4: Un-obfuscated strings identifying the Remcos server component

Numerous commands that the server can carry out can also be seen in plain text.

```
}
v171 = "filemgr";
if ( std::operator==(&v196) )
{
  v25 = sub_401289(&v197, 1u);
  std::basic_string<char,std::char_traits<char>,std::allocator<char>>::b
    &v168,
    v25);
  sub_40C75C(v168);
  goto LABEL_96;
}
v171 = "downloadfromurltofile";
if ( std::operator==(&v196) )
{
  v26 = sub_401289(&v197, 2u);
  v27 = std::basic_string<char,std::char_traits<char>,std::allocator<cha
  sub_40714C(&v192, 48, v27);
  v171 = 0;
  v170 = 0;
  v169 = std::basic_string<char,std::char_traits<char>,std::allocator<ch
  v28 = sub_401289(&v197, 1u);
  v29 = std::basic_string<char,std::char_traits<char>,std::allocator<cha
  URLDownloadToFileA(0, v29, v169, v170, v171);
  v171 = 1;
  v170 = 0;
  v169 = 0;
  v30 = std::basic_string<char,std::char_traits<char>,std::allocator<cha
  ShellExecuteA(0, "open", v30, v169, v170, v171);
  v31 = &v192;
EL_16:
  std::basic_string<char,std::char_traits<char>,std::allocator<char>>::~
  goto LABEL_96;
}
v171 = "downloadfromlocaltofile";
if ( std::operator==(&v196) )
{
  v32 = sub_401289(&v197, 1u);
```

Figure 5: Snippet of some commands

Figuring out all the commands through code analysis is tedious work. Fortunately, their website allows anyone to download a stripped down version of the Remcos client for free.

Remcos v.1.7.3 and its Capabilities

The Remcos Client has five main tabs with different specific functions. Although most of the parameters are disabled in the free version, we were able to simulate its client-server connection.

The *Connections* Tab is where all the active connections can be monitored. Each entry contains some basic information about the installed server component and the infected system. This is also the main tab for sending commands to the infected system. The image below shows the list of commands that can be executed in the infected system. It illustrates how much control the attacker can gain over an infected system. Most of them are fairly common with RAT applications, and as usual some of the commands may lean more towards intrusive spying than consented monitoring.

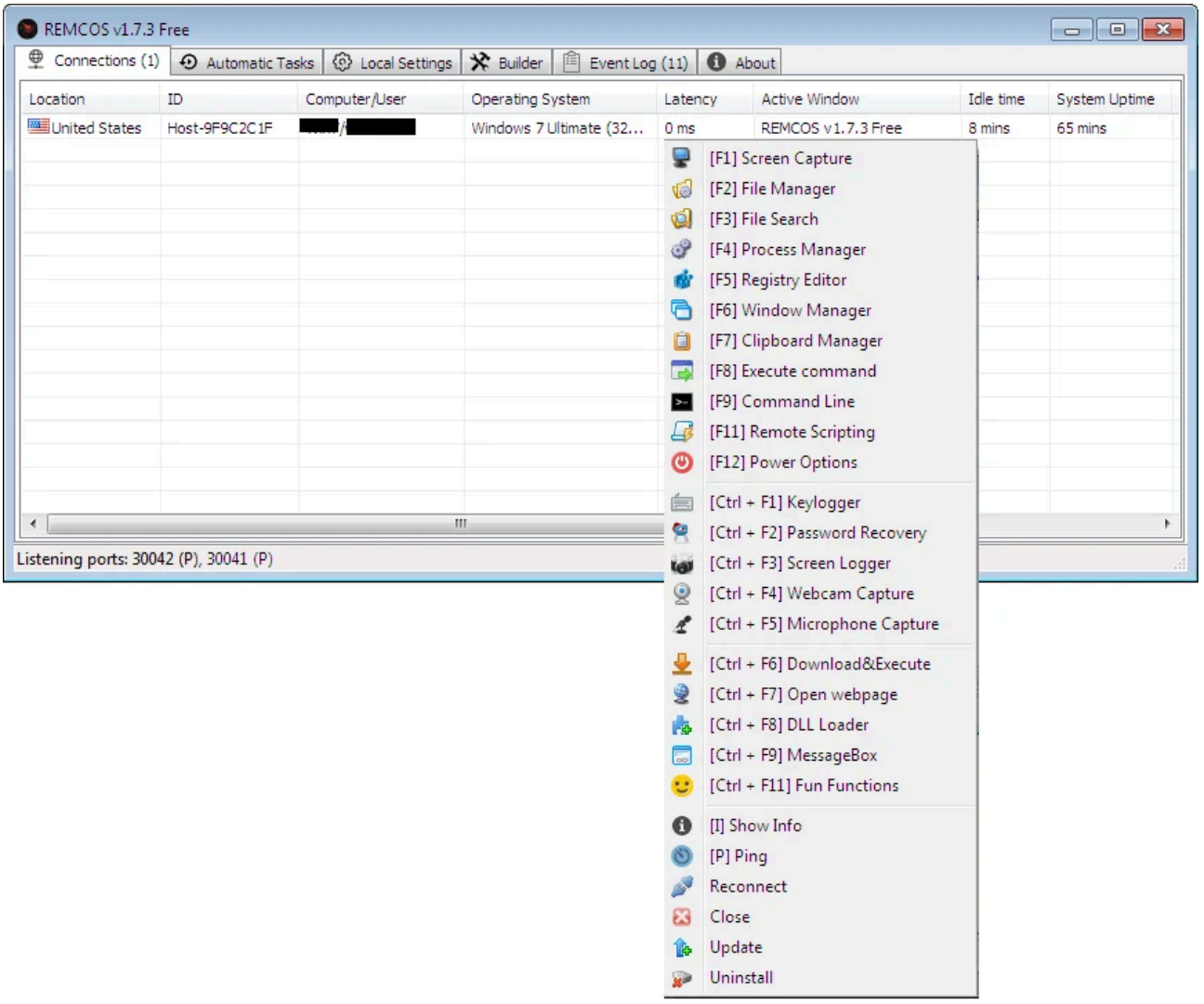

Figure 6: REMCOS command list

Automatic Tasks is probably the most interesting feature of Remcos, as we haven't seen anything like it on other RATs. This feature configures the server component to automatically execute functions without any manual action from the client once a connection has been established. This makes it easy and convenient to create an infiltrate-exfiltrate-exit scheme without any trigger from the attacker, which is just how a common spyware or malware downloader behaves.

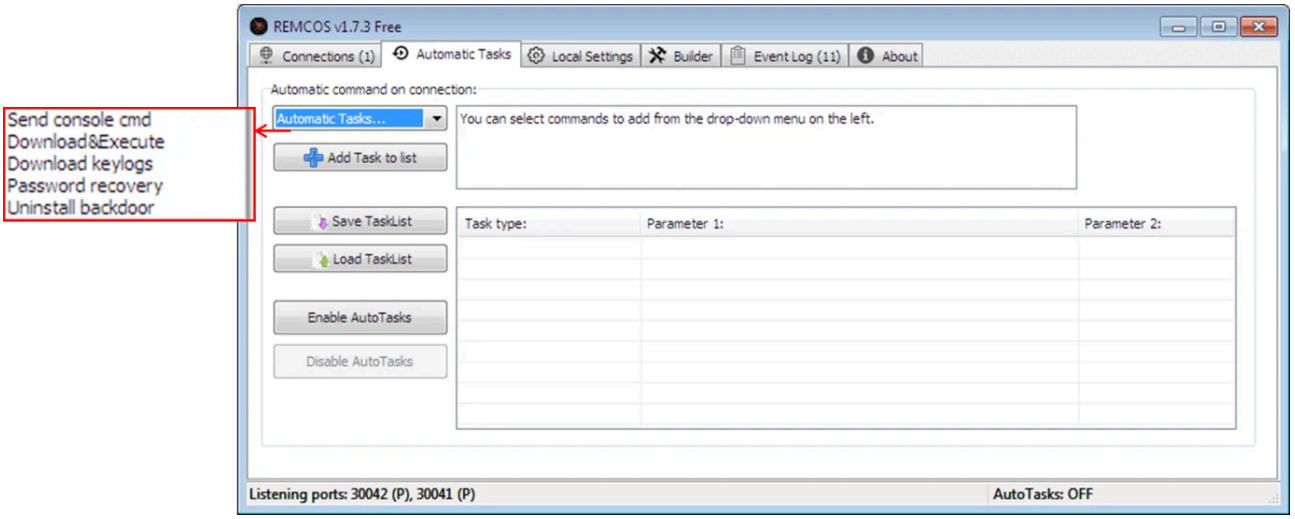

Figure 7: Automatic Tasks tab

The *Local Settings* tab consists of settings for the client side. Ports where the client machine waits for a connection from its servers are set here, together with the passwords to be used. Since Remcos uses the password for encryption, the listening port and the connecting server should have the same passwords for a successful connection. So basically, the password is used for both authentication and network traffic encryption.

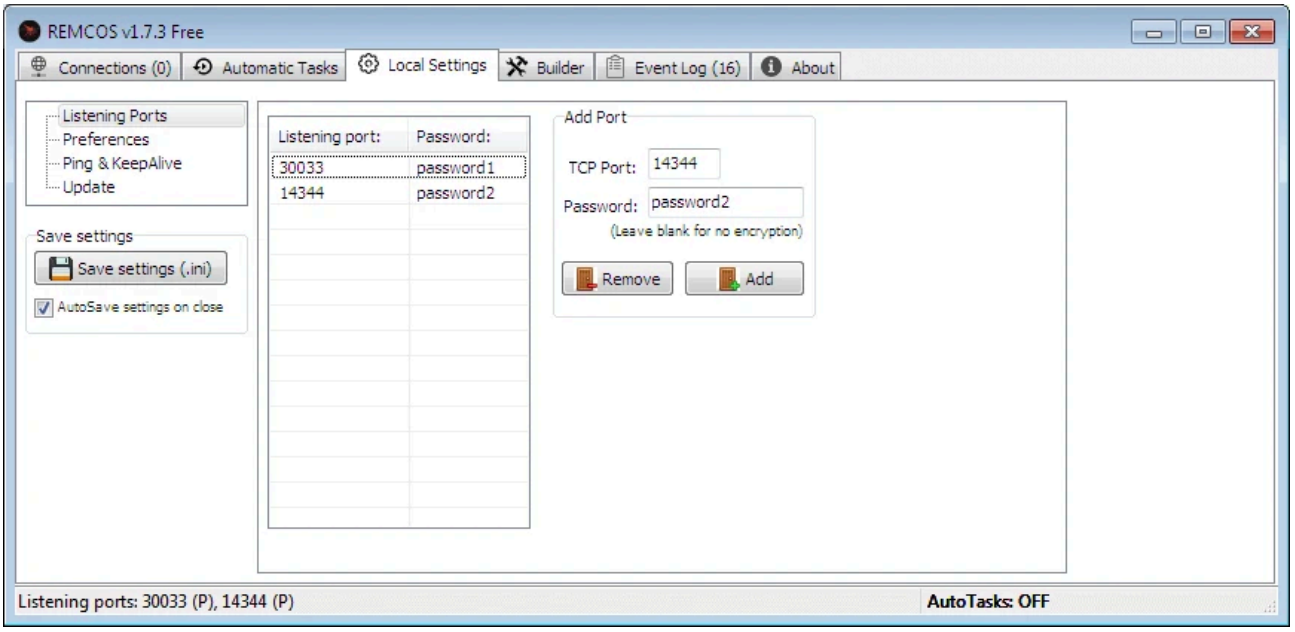

Figure 8: Local Settings tab

Remcos uses a simple RC4 algorithm, using the password as the key to encrypt and decrypt network traffic between its client and server.

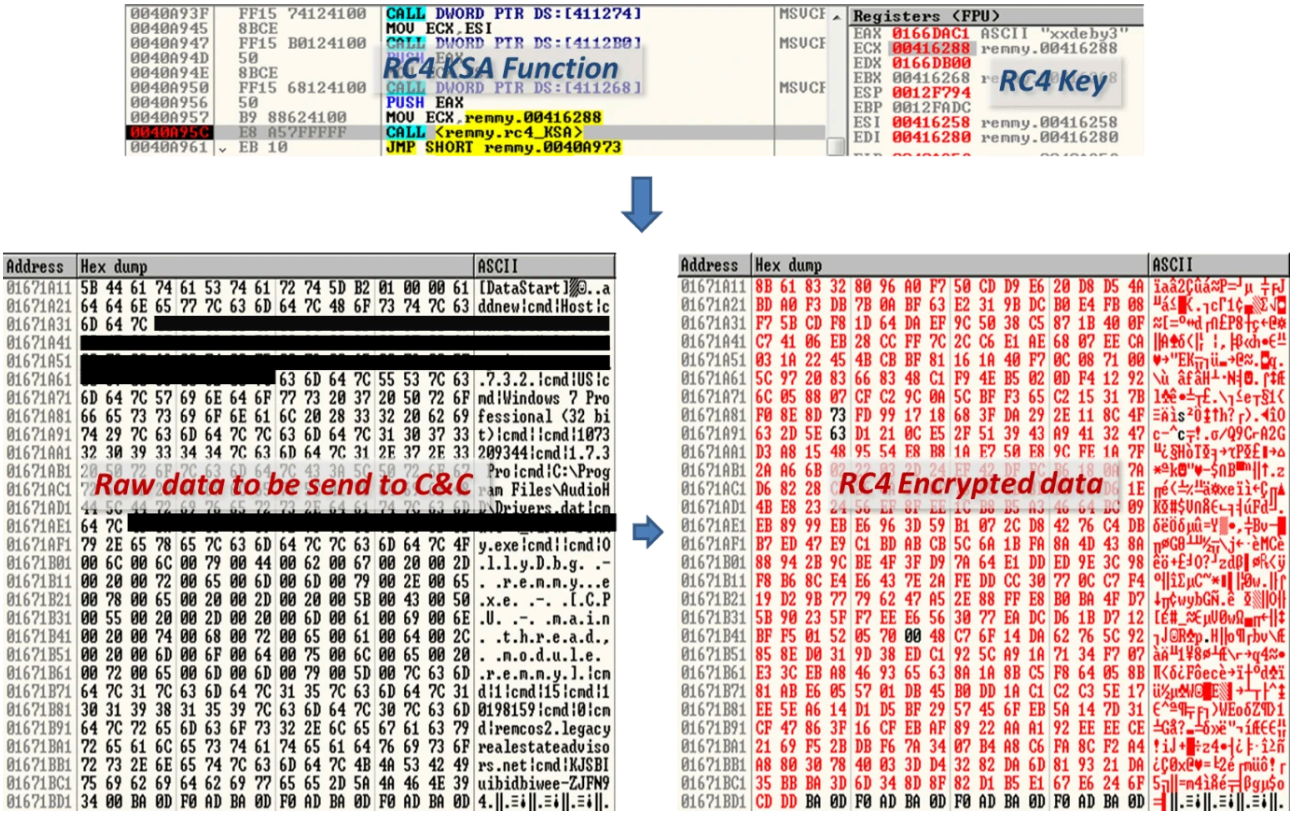

Figure 9: Uses RC4 algorithm to encrypt network traffic

The *Builder* tab is where the parameters of the created server binary can be customized. It can be divided into several sub-sections, as shown in the image below.

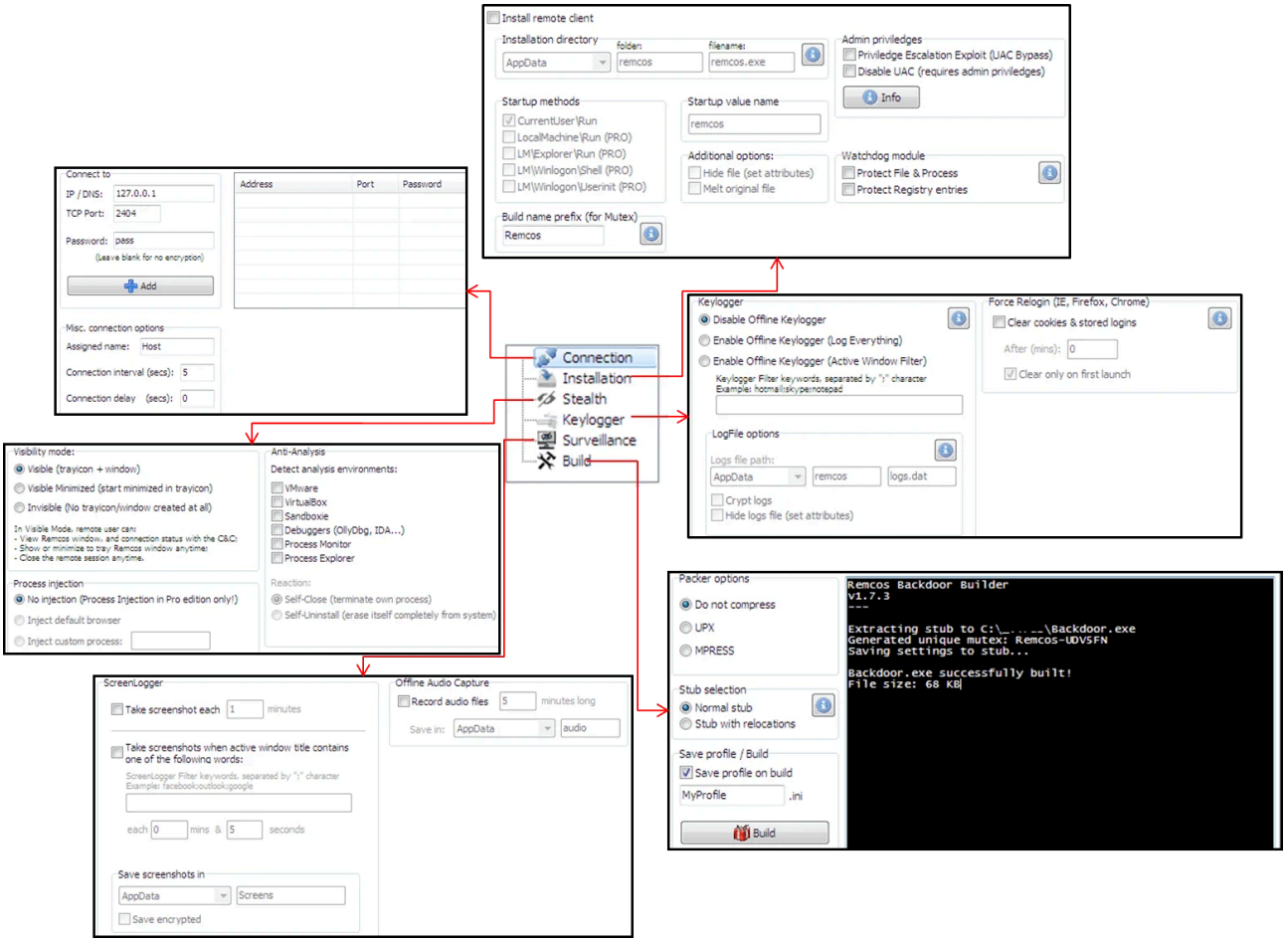

Figure 10: Builder tab sub-sections

Connection – sets the client IP addresses and ports where the server connects to upon installation. It also allows a password to be set for authentication and encryption.

Installation – configures the installation path, autorun registries, and a watchdog module that prevents termination of the process and deletion of its files and registries. Also included in this section is the setting for having its own UAC bypass, which we suspected to exist earlier in our article.

So, it is possible that the attacker only used the document macro as a template to download and execute the binary, and never intended to use the script's UAC bypass since the server binary itself already has the same function. In fact, it uses the same UAC bypass technique, but this time with an added routine to revert the modified registry after gaining privilege. This is logical, because not restoring the registry can produce system errors that can cause suspicion from the user every time a .msc file needs to be opened.

Stealth – this section dictates whether the server should appear on the system's tray icon. It also includes the settings for some basic anti-analysis/anti-sandbox routines and an option to hide the process through injection.

Keylogger – this includes the usual parameters for a basic keylogger function. Interestingly enough, though, it can also provide the server component with a function to remove browser cookies and stored passwords. The hope is that that the user will have to re-type their passwords when logging in to websites and they can be captured using the keylogger.

Surveillance – gives the server an option to take periodic screenshots of the system or when specific windows are active. It also features audio capture, which can be saved locally for later retrieval.

Build – gives the option to pack the server binary using UPX and MPRESS.

The *Event Log* displays connection logs with the server, along with some information regarding the client's status (updates, ports, etc.).

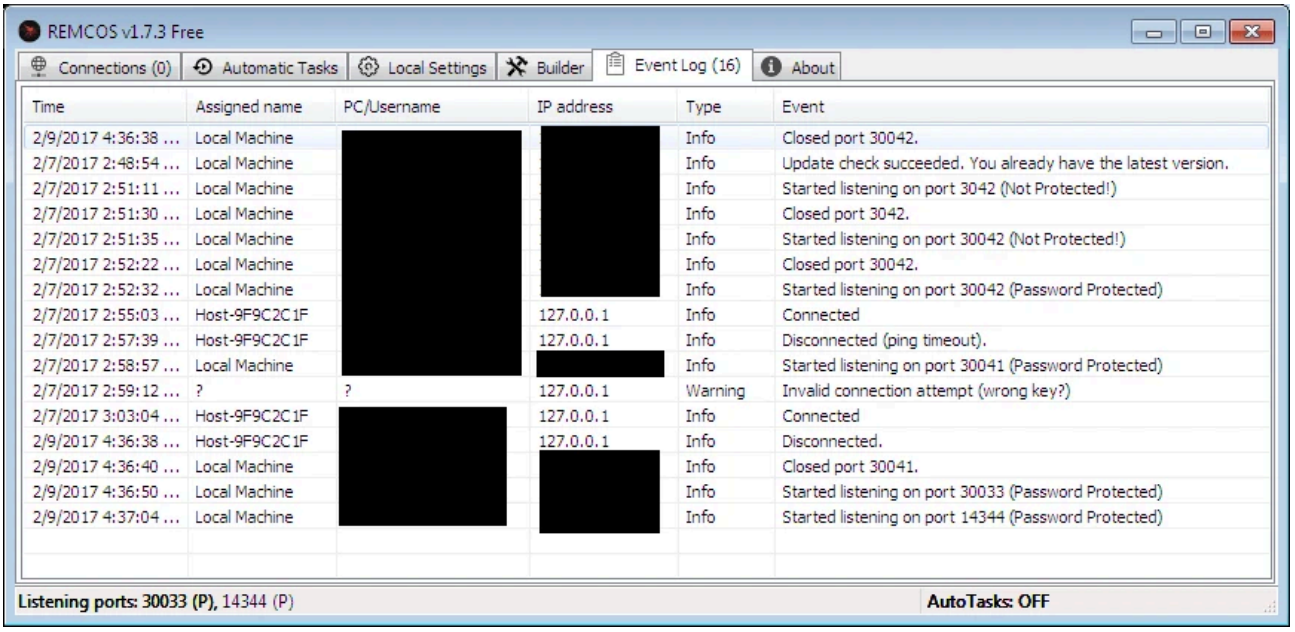

Figure 11: Event Logs tab

The *About* tab contains acknowledgements and some promotions on other products that have been developed by an author named *Viotto*.

Figure 12: About tab

Conclusion

This article proves once again that one does not have to be an expert to launch fairly sophisticated malware attacks. More and more applications like Remcos are being released publicly, luring new perpetrators with their easy usage. And all it takes to be infected by one are a few clicks.

As for many RAT authors, the developer discourages malicious usage of the tool through a license ban if reported. This in most cases is nothing but a false shield to guard them liability when the thin veil of its being an administration tool is removed and it is exposed as a full-blown malware builder.

-= FortiGuard Lion Team =-

Samples (SHA256)

fc0fa7c20adf0eaf0538cec14e37d52398a08d91ec105f33ea53919e7c70bb5a - *W32/Remcos.A!tr*

8710e87642371c828453d59c8cc4edfe8906a5e8fdfbf2191137bf1bf22ecf81 - *W32/Remcos.A!tr*

8e6daf75060115895cbbfb228936a95d8fb70844db0f57fe4709007a11f4a6bb - *WM/Agent.9BF1!tr.dldr*

a58a64fce0467acbcaf7568988afc6d2362e81f67fc0befd031d3a6f3a8a4e30 - *WM/Agent.9BF1!tr.dldr*

IOC

Download URL:

legacyrealestateadvisors[.]net/brats/remmy.exe

C&C:

remcos2.legacyrealestateadvisors[.]net

remcos.legacyrealestateadvisors[.]net

Added Files (paths can be changed in the builder):

%temp%\remmy.exe – *copy of server*

%ProgramFiles%\AudioHD\Drivers.dat – *keylog data*

%ProgramFiles%\AudioHD\AudioHD.exe or %ProgramFiles%\SvchostHD\svchost.exe – *copy of server*

Added Registries:

Key: HKCU \Software\Microsoft\Windows\CurrentVersion\Run

Value Name: SvchostHD

Data: %ProgramFiles%\SvchostHD\svchost.exe

Or

Key: HKCU \Software\Microsoft\Windows\CurrentVersion\Run

Value Name: AudioHD

Data: %ProgramFiles%\ AudioHD\AudioHD.exe

Key: HKCU \Software\-

Value Name: EXEpath
