Hello! My name is Dtrack

securelist.com/my-name-is-dtrack/93338/

Authors

Expert Konstantin Zykov

Our investigation into the Dtrack RAT actually began with a different activity. In the late summer of 2018, we discovered ATMDtrack, a piece of banking malware targeting Indian banks. Further analysis showed that the malware was designed to be planted on the victim's ATMs, where it could read and store the data of cards that were inserted into the machines. Naturally, we wanted to know more about that ATM malware, so we used YARA and Kaspersky Attribution Engine to uncover more interesting material: over 180 new malware samples of a spy tool that we now call Dtrack.

All the Dtrack samples we initially found were dropped samples, as the real payload was encrypted with various droppers — we were able to find them because of the unique sequences shared by ATMDtrack and the Dtrack memory dumps. After that, it got very interesting, because once we decrypted the final payload and used Kaspersky Attribution Engine again, we saw similarities with the DarkSeoul campaign, dating back to 2013 and attributed to the Lazarus group. It seems that they reused part of their old code to attack the financial sector and research centers in India. According to our telemetry, the last activity of DTrack was detected in the beginning of September 2019.

Technical details

The dropper has its encrypted payload embedded as an overlay of a PE file as extra data that will never be used in normal execution steps. Its decryption routine, part of an executable physical patch, begins somewhere between the *start()* and *WinMain()* functions. A fun fact is that the malware authors embedded their malicious code into a binary that was a harmless executable. In some cases, it was the default Visual Studio MFC project, but it could be any other program.

The decrypted overlay data contains the following artifacts:

- an extra executable;
- process hollowing shellcode;
- a list of predefined executable names, which the malware uses as a future process name.

After decryption of the data, the process hollowing code is started, taking the name of the process to be hollowed as an argument. The name comes from the predefined list found within the decrypted overlay. All the names come from the %SYSTEM32% folder, as you can see in the decrypted file list below.

- fontview.exe
- dwwin.exe
- wextract.exe
- runonce.exe
- grpconv.exe
- msiexec.exe
- rasautou.exe
- rasphone.exe
- extrac32.exe
- mobsync.exe
- verclsid.exe
- ctfmon.exe
- charmap.exe
- write.exe
- sethc.exe
- control.exe
- presentationhost.exe
- napstat.exe
- systray.exe

- mstsc.exe
- cleanmgr.exe

What is inside the dropper?

After execution, the target of the process hollowing is suspended until its memory is overwritten with the decrypted executable payload from the dropper overlay. After this, the target process resumes.

The droppers contain a variety of executables, all of these intended for spying on the victim. Below is an incomplete functionality list for the various Dtrack payload executables found:

- keylogging,
- retrieving browser history,
- gathering host IP addresses, information about available networks and active connections,
- listing all running processes,
- listing all files on all available disk volumes.

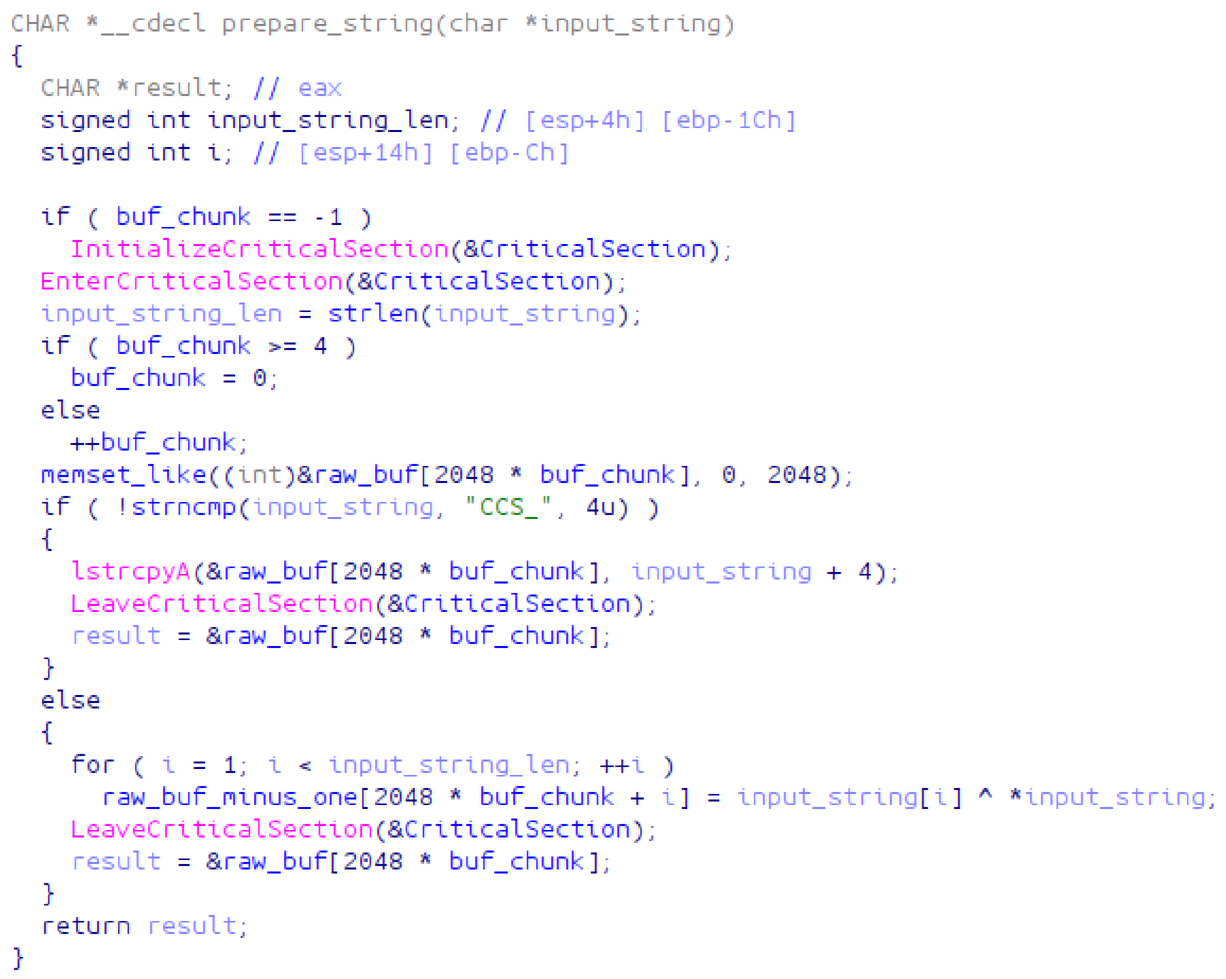
At this point, the design philosophy of the framework becomes a bit unclear. Some of the executables pack the collected data into a password protected archive and save it to the disk, while others send the data to the C&C server directly.

Aside from the aforementioned executables, the droppers also contained a remote access Trojan (RAT). The RAT executable allows criminals to perform various operations on a host, such as uploading/downloading, executing files, etc. For a full list of operations, see the table below.

command id	description
1003	upload a file to the victim’s computer
1005	make target file persistent with auto execution on the victim’s host start
1006	download a file from the victim’s computer
1007	dump all disk volume data and upload it to a host controlled by criminals
1008	dump a chosen disk volume and upload it to a host controlled by criminals
1011	dump a chosen folder and upload it to a host controlled by criminals
1018	set a new interval timeout value between new command checks
1023	exit and remove the persistence and the binary itself
default	execute a process on the victim’s host

Dtrack and ATMDTrack malware similarities

ATMDTrack is a subset of the DTrack family. They naturally look different despite their similarities. For example, Dtrack's payload is encrypted within a dropper—unlike the ATMDTrack samples, which were not encrypted at all. But after decrypting the Dtrack payload, it becomes clear that the developers are the same group of people: both projects have the same style and use the same implemented functions. The most obvious function they have in common is the string manipulation function. It checks if there is a *CCS_* substring at the beginning of the parameter string, cuts it out and returns a modified one. Otherwise, it uses the first byte as an XOR argument and returns a decrypted string.



```
CHAR *__cdecl prepare_string(char *input_string)
{
  CHAR *result; // eax
  signed int input_string_len; // [esp+4h] [ebp-1Ch]
  signed int i; // [esp+14h] [ebp-Ch]

  if ( buf_chunk == -1 )
    InitializeCriticalSection(&CriticalSection);
  EnterCriticalSection(&CriticalSection);
  input_string_len = strlen(input_string);
  if ( buf_chunk >= 4 )
    buf_chunk = 0;
  else
    ++buf_chunk;
  memset_like((int)&raw_buf[2048 * buf_chunk], 0, 2048);
  if ( !strncmp(input_string, "CCS_", 4u) )
  {
    lstrcpyA(&raw_buf[2048 * buf_chunk], input_string + 4);
    LeaveCriticalSection(&CriticalSection);
    result = &raw_buf[2048 * buf_chunk];
  }
  else
  {
    for ( i = 1; i < input_string_len; ++i )
      raw_buf_minus_one[2048 * buf_chunk + i] = input_string[i] ^ *input_string;
    LeaveCriticalSection(&CriticalSection);
    result = &raw_buf[2048 * buf_chunk];
  }
  return result;
}
```

Functions common to the two families (the functions/arguments were named by the researchers)

Conclusions

When we first discovered ATMDtrack, we thought we were just looking at another ATM malware family, because we see new ATM malware families appearing on a regular base. However, this case proved once again that it is important to write proper YARA rules and have a solid working attribution engine, because this way you can uncover connections with malware families that have appeared in the past. One of the most memorable examples of this was the WannaCry attribution case. Now we can add another family to the Lazarus group's arsenal: ATMDtrack and Dtrack.

The vast amount of Dtrack samples that we were able to find shows that the Lazarus group is one of the most active APT groups in terms of malware development. They continue to develop malware at a fast pace and expand their operations. We first saw early samples of this malware family in 2013, when it hit Seoul. Now, six years later, we see them in India, attacking financial institutions and research centers. And once again, we see that this group uses similar tools to perform both financially-motivated and pure espionage attacks.

To succeed in spying, the criminals should be able to gain at least partial control over the internal network. This means that the target organizations may have a number of security issues, such as:

- weak network security policies,
- weak password policies,
- lack of traffic monitoring.

We therefore advise the companies to:

- tighten their network and password policies,
- use traffic monitoring software, such as Kaspersky Anti Targeted Attack Platform (KATA),
- use antivirus solutions.

IoCs

- 8f360227e7ee415ff509c2e443370e56
- 3a3bad366916aa3198fd1f76f3c29f24
- F84de0a584ae7e02fb0ffe679f96db8d

- ATM
- Dropper
- Financial malware
- Lazarus
- Malware Descriptions
- RAT Trojan

Authors

Expert Konstantin Zykov

Hello! My name is Dtrack
