DanaBot's Latest Move: Deploying Latrodectus

esentire.com/blog/danabots-latest-move-deploying-icedid

What We Do

eSentire MDR for Microsoft

Visibility and response across your entire Microsoft security ecosystem.

Learn More →

Resources

TRU Intelligence Center

Our Threat Response Unit (TRU) publishes security advisories, blogs, reports, industry publications and webinars based on its original research and the insights driven through proactive threat hunts.

EXPLORE RESOURCES →

Company

ABOUT ESENTIRE

eSentire is The Authority in Managed Detection and Response Services, protecting the critical data and applications of 2000+ organizations in 80+ countries from known and unknown cyber threats. Founded in 2001, the company's mission is to hunt, investigate and stop cyber threats before they become business disrupting events.

About Us →

Leadership →

Careers →

EVENT CALENDAR

Feb

01

VeraLogics Dine & Dash

Feb

13

February TRU Intelligence Briefing

Feb

13

SIM Houston Roundtable

View Calendar →

Partners

PARTNER PROGRAM

LEARN MORE →

Apply to become an e3 ecosystem partner with eSentire, the Authority in Managed Detection and Response.

APPLY NOW →

Login to the Partner Portal for resources and content for current partners.

LOGIN NOW →

Get Started

Want to learn more on how to achieve Cyber Resilience?

TALK TO AN EXPERT

Adversaries don't work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire's Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here's the latest from our TRU Team…

What did we find?

In December 2023, this blog post was revised based on insights from Proofpoint's researcher, known as @Myrtus0x0. The malware under investigation has been identified as 'Latrodectus', which is believed to have been developed by the creators of IcedID.

In early November 2023, the eSentire Threat Response Unit (TRU) detected the presence of DanaBot, a sophisticated banking Trojan renowned for its ability to pilfer banking credentials, personal information, and hVNC (hidden Virtual Network Computing) feature (Figure 1).

This malware was being employed to deliver IcedID, a banking Trojan that has been active since 2017 and is widely recognized for its various capabilities. Notably, since 2020, IcedID has been linked to ransomware attacks, including those involving Egregor, Maze, and Conti.

Stealer (FF,OP,CH,IE,EG).
Post grabber (EG,FF,OP,CH,IE)
HTML injections with extended Zeus format including internal variables and bot information.
HVNC is a hidden desktop.
Video recording, processes, sites.
Keyboard spy.
Interception of the Clipboard.
Redirects of web requests.
Blocking web requests.
Jabber notification system for events, sites, processes, automatic activation of the HVNC function.
Offline file management system. (wallet grabber).
Screen view, cmd, process management.
Restoring information about proxy servers via TOR.

Сервер / Server

Система генерации билдов.
Автокрипт. (NO/FUD)
Система построения цепочки прокси (+Tor) автопроверка. Софт для проксирования.
Система API для подключения ваших крипторов, баз данных.
Встроенный Фаервол.

Build generation system.
Autocrypt. (NO/FUD)
The system of building a proxy chain (+Tor) auto-check. Software for design.
API system for connecting your cryptors, databases.
Built-in Firewall.

Программа клиент для управления / Client

Панель для работы с данными в виде exe приложения для windows.
Возможность управления ботами в офлайн режиме.
Система экспорта слепка бота (Браузер, куки, сохранные пароли, версия операционной системы).
Быстрая и удобная обработка большого объема данных, парсер данных.
Многоформатный Экспорт данных.
Режим чтения логов без возможности управлять системой (для сотрудников).

A panel for working with data in the form of an exe application for windows.
The ability to manage bots in offline mode.
The bot's impression export system (Browser, cookies, saved passwords, operating system version).
Fast and convenient processing of a large amount of data, data parser.
Multi-format Data export.
Log reading mode without the ability to manage the system (for employees).

Figure 1: DanaBot advertisement on a Russian hacking forum

In a recent case, we assess with high confidence that the initial infection for DanaBot occurred via a drive-by download. The user was likely searching for a Webex installer and visited an imposter website serving the payload. The archive payload is named Webex.zip (MD5: 4be85751a07081de31f52329c2e2ddc8).

The archive contains the following files:

- rash.docx (IDAT Loader encrypted file), MD5: 34b87976172e911e3e2ed6007252e7dc

- sqlite3.dll – malicious side-loaded DLL, MD5: 4ca6db064effc1730299a0f20531e49c
- webex.exe – legitimate binary, MD5: 1f166f5c76eb155d44dd1bf160f37a6a

Upon execution of webex.exe, it will side-load the malicious DLL (sqlite3.dll), decrypt and decompress the contents of rash.docx file, perform injection into explorer.exe via Process Doppelgänging, and decrypt and run the final payload. In our case, it's DanaBot (MD5: 6ad1d4e1ca3f1784840364700f5a8a14).

We have observed DanaBot dropping the following files on the infected system under %TEMP% folder:

- 10608194856200.exe (MD5: 0d0c437a39787127fc0fbf19efc747ab), the file is, what we assess, an IcedID VNC module
- c5cfe172.dll, IcedID loader (MD5: 350915536540a76d44ce12dc03450424)

Upon execution of the IDAT loader, two folders are created under %AppData%:

- DownloadWordpadISR (folder that contains rash.docx and sqlite3.dll)
- Custom_update (folder that contains IcedID payloads Update_* (for example, Update_88d58563) and update_data.dat)

The persistence for DanaBot is created via Startup folder (T1547.001) for webex.exe binary.

IcedID Technical Analysis

After the IcedID payload decryption, it creates a copy of itself under *"%AppData%\Custom_update\ Update_{8-hexidecimal-characters}"*. The 8 hexadecimal characters are determined by the function in Figure 2.

```
 3  __int64 result; // rax
 4  unsigned int i; // [rsp+20h] [rbp-18h]
 5
 6  *(_DWORD *)calculated_seed_val = mw_seed(volume_serial_number);
 7  *(_WORD *)(calculated_seed_val + 4) = mw_seed(volume_serial_number);
 8  result = mw_seed(volume_serial_number);
 9  *(_WORD *)(calculated_seed_val + 6) = result;
10  for ( i = 0; i < 8; ++i )
11  {
12    *(_BYTE *)(calculated_seed_val + i + 8) = mw_seed(volume_serial_number);
13    result = i + 1;
14  }
15  return result;
16 }
```

Figure 2: Hexadecimal value generation

The payload retrieves the volume serial number of the infected machine via GetVolumeInformationW API and multiplies the result with the seed value 0x19660D. The returned result is then used as a part of the DLL filename appended after *"Update_"* as 8

hexadecimal characters.

The function then proceeds and enters the loop where it performs the multiplication with the seed value with the result of each seeded value returned from the mw_seed function; it then grabs the first byte from each calculated result and builds a 14-byte unique HWID string that is sent to C2.

IcedID uses a CRC-32 hashing algorithm to calculate the hashes for the APIs used in the binary (Figures 3-4).

```
 8
 9  if ( !dword_1800101A8 )
10  {
11    for ( i = 0; i < 0x100; ++i )
12    {
13      v5 = i;
14      for ( j = 8; j; --j )
15      {
16        if ( (v5 & 1) != 0 )
17          v5 = (v5 >> 1) ^ 0xEDB88320;
18        else
19          v5 >>= 1;
20      }
21      dword_18000FDA8[i] = v5;
22    }
23    dword_1800101A8 = 1;
24  }
25  v6 = -1;
26  for ( k = 0; k < a2; ++k )
27    v6 = dword_18000FDA8[(unsigned __int8)(v6 ^ *(_BYTE *)(a1 + k))] ^ (v6 >> 8);
28  return ~v6;
29 }
```

Figure 3: CRC-32 API hashing function

```
288  v21 = 0x8197004C;
289  v22 = &unk_18000FD50;
290  v23 = (__int64 *)&Process32FirstW;
291  v24 = 645722533;
292  v25 = &unk_18000FD50;
293  v26 = (__int64 *)&Process32First;
294  v27 = 0xBC6B67BF;
295  v28 = &unk_18000FD50;
296  v29 = (__int64 *)&Process32NextW;
297  v30 = 0x28ED5C0;
298  v31 = &unk_18000FD50;
299  v32 = (__int64 *)&Process32Next;
```

Figure 4: CRC-32 calculated API hashes

The string decryption (Figure 5) is performed based on the following algorithm:

- The function initializes using the first 4 bytes derived from the encrypted string.
- Within prng_gen function, it generates a series of pseudo-random values based on the first 4-bytes derived from the encrypted string.
- For each byte in a certain range, it performs a bitwise XOR with the pseudo-random value and a byte from the offset location in the encrypted string.

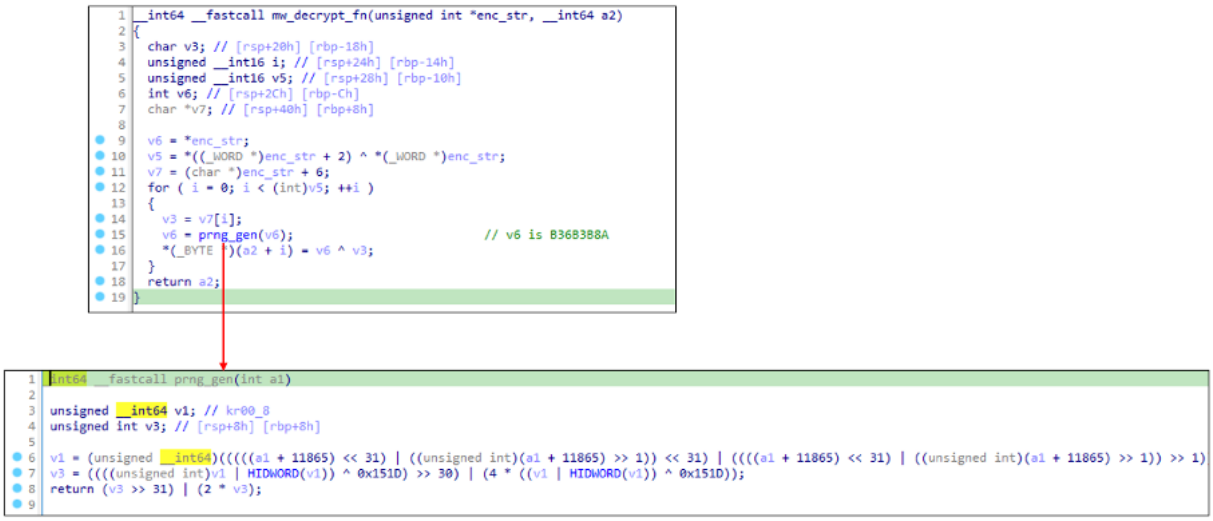

Figure 5: String decryption function

We wrote the script to decrypt the strings with IDAPython. You can access the script here.

The decrypted strings can be accessed here.

IcedID creates the hardcoded mutex “runnung". If the payload fails to create a mutex or if the mutex already exists (indicated by the error code 183, which typically means ERROR_ALREADY_EXISTS), then the payload enters an infinite loop delay using NtDelayExecution (1000 milliseconds of delay) (Figure 6).

This prevents multiple instances of infections on the same infected machine.

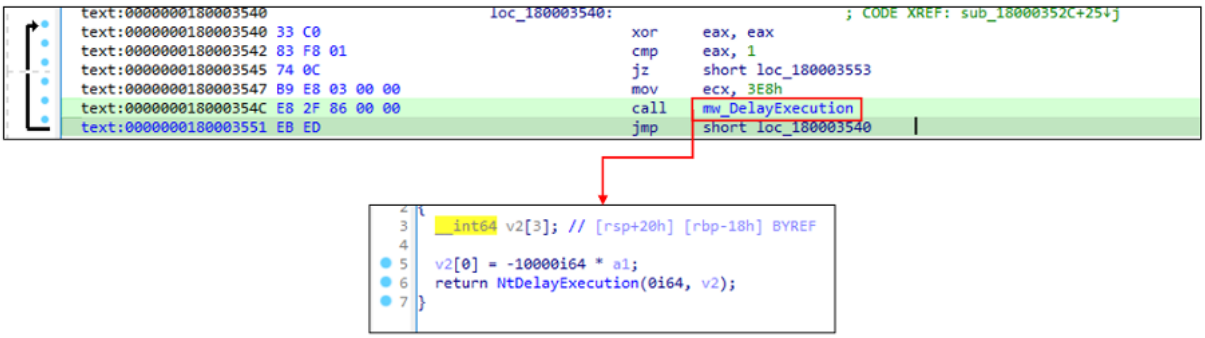

Figure 6: IcedID enters an infinite loop of delays if the mutex already exists

The campaign ID is generated using the hardcoded string in the binary; in our binary, it’s *“Novik”*, and FNV hashing algorithm.

```
__int64 __fastcall mw_fnv(char *hardcoded_str, __int64 str_len)
{
  unsigned int v3; // [rsp+0h] [rbp-18h]
  char *i; // [rsp+8h] [rbp-10h]

  v3 = 0x811C9DC5;
  for ( i = hardcoded_str; i < &hardcoded_str[str_len]; ++i )
    v3 = 0x1000193 * (*i ^ v3);
  return v3;
}
```

Figure 7: FNV hashing algorithm

Here is the implementation of the algorithm in Python:

```
def mw_fnv(input_str):
    v3 = 0x811C9DC5
    for char in input_str:
        v3 = (v3 ^ ord(char)) * 0x1000193
        v3 &= 0xFFFFFFFF
    return v3
fnv_hash = mw_fnv("Novik") # input your hardcoded string here
print(fnv_hash)
```

Upon successful infection, IcedID runs the following reconnaissance commands on the infected host:

- C:\Windows\System32\cmd.exe /c ipconfig /all
- C:\Windows\System32\cmd.exe /c systeminfo
- C:\Windows\System32\cmd.exe /c nltest /domain_trusts
- C:\Windows\System32\cmd.exe /c nltest /domain_trusts /all_trusts
- C:\Windows\System32\cmd.exe /c net view /all /domain
- C:\Windows\System32\cmd.exe /c net view /all
- C:\Windows\System32\cmd.exe /c net group "Domain Admin”
- C:\Windows\System32\wbem\wmic.exe /Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get * /Format:List
- C:\Windows\System32\cmd.exe /c net config workstation
- C:\Windows\System32\cmd.exe /c wmic.exe /node:localhost /namespace:\\root\SecurityCenter2 path AntiVirusProduct Get DisplayName | findstr /V /B /C:dis
- C:\Windows\System32\cmd.exe /c whoami /groups
- C:\Windows\System32\cmd.exe
- C:\Windows\System32\cmd.exe

The results then are converted into base64-encoded strings and appended to the following tags accordingly:

- &ipconfig=
- &systeminfo=
- &domain_trusts=
- &domain_trusts_all=
- &net_view_all_domain=
- &net_view_all=
- &net_group=
- &net_config_ws=
- &net_wmic_av=
- &whoami_group=

Figure 8 shows the function responsible for the following:

- Decrypts and sets HTTP headers (C*ontent-Type: application/x-www-form-urlencoded*).
- Determines the request method (POST or GET) based on the input parameter **a2**, decrypts the relevant method string, and prepares it for use.
- Calls *HttpOpenRequestA* with parameters including the request method, URL, and other settings. The request is opened using a handle provided by *InternetOpenW*.
- Checks if HttpOpenRequestA successfully created a request handle.
    - If it's a POST request, it calculates the length of the request data and headers, then sends the HTTP request with *HttpSendRequestA* using these lengths and the base64-encoded data.
    - If it's a GET request, it sends the request without additional data.

```
54  if ( !ptr_HttpOpenRequestA )
55    return 0i64;
56  if ( a6 == 4 )
57  {
58    v12 = 17280;
59    ((void (__fastcall *)(__int64, __int64, int *, __int64))InternetSetOptionA)(ptr_HttpOpenRequestA, 31i64, &v12, 4i64);
60  }
61  if ( a2 )
62  {
63    ptr_get_str_len = mw_get_str_len(base64_enc_str);
64    v7 = mw_get_str_len((__int64)&unk_18000F350);
65    ptr_HttpSendRequestA = ((__int64 (__fastcall *)(__int64, void *, _QWORD, __int64, int))HttpSendRequestA)(
66                             ptr_HttpOpenRequestA,
67                             &unk_18000F350,
68                             v7,
69                             base64_enc_str,
70                             ptr_get_str_len);
71  }
72  else
73  {
74    ptr_HttpSendRequestA = ((__int64 (__fastcall *)(__int64, _QWORD, _QWORD, _QWORD, _DWORD))HttpSendRequestA)(
75                             ptr_HttpOpenRequestA,
76                             0i64,
77                             0i64,
78                             0i64,
79                             0);
80  }
81  if ( ptr_HttpSendRequestA )
82    return ptr_HttpOpenRequestA;
```

Figure 8: HTTP Request Handler Function

The payload enumerates through the list of running processes using APIs such as CreateToolhelp32Snapshot, Process32First, and Process32Next and appends the results to the following tags:

- &proclist=
- “pid":
- "proc":
- "subproc":

The persistence is achieved via the scheduled task named “Updater”. The task runs at every log on with the following command:

> rundll32.exe "C:\Users\
> <username>\AppData\Roaming\Custom_update\Update_88d58563.dll", scab

Previously, we mentioned IcedID deploying the VNC module. There are a few interesting strings in the payload that we observed:

- {%0.8X-%0.4X-%0.4X-%0.4X-%0.4X%0.8X}
- %ProgramFiles%\
- gw@SET TO TOP
- %ProgramData%\
- %LOCALAPPDATA%\
- {"dev":[
- %sProfile %u\
- FOREGRAUND
- %APPDATA%\
- %ProgramFiles(x86)%\
- /NOUACCHECK
- xpChrome_WidgetWin_
- NEW FOREGRAUND
- aaa_11.02_mmm
- hdesk

What did we do?

Our team of 24/7 SOC Cyber Analysts detected malicious network connections originating from the rundll32.exe process, isolated the affected machine, and informed the impacted customer.

What can you learn from this TRU Positive?

- The use of drive-by downloads for initial infection shows the effectiveness of this method for deploying malware, emphasizing the risk associated with unverified downloads.
- The execution of a legitimate binary to side-load a malicious DLL highlights advanced techniques used by attackers to evade detection.

- IcedID's approach to preventing multiple infections on the same machine using a hardcoded mutex and entering an infinite loop if the mutex exists showcases a method to avoid detection and potential interference with other malware.
- The use of the FNV hashing algorithm to generate campaign IDs based on hardcoded strings.
- The execution of various system commands post-infection for reconnaissance purposes underlines the importance of monitoring command line activity on endpoints.Top of Form

Recommendations from our Threat Response Unit (TRU) Team:

- Prioritize application installations from your organization's library of approved applications (if implemented).
- Treat files downloaded from the Internet with the same vigilance as those delivered through email. Assume files are potentially hostile regardless of the path that got you there. Remember, a website hosting software advertised on a trusted search engine does not inherit that trust.
- Encouraging good cybersecurity hygiene among your users by using Phishing and Security Awareness Training (PSAT) when downloading software from the Internet.
- Protect endpoints against malware by:
    - Ensuring antivirus signatures are up-to-date.
    - Using a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) tool to detect and contain threats.

Indicators of Compromise

Name	Indicator
Webex.zip	4be85751a07081de31f52329c2e2ddc8).
rash.docx	34b87976172e911e3e2ed6007252e7dc
sqlite3.dll	4ca6db064effc1730299a0f20531e49c
10608194856200.exe	0d0c437a39787127fc0fbf19efc747ab),
c5cfe172.dll	350915536540a76d44ce12dc03450424)
DanaBot	6ad1d4e1ca3f1784840364700f5a8a14).

IcedID C2	arsimonopa[.]com/live
IcedID C2	lemonimonakio[.]com/live
IcedID VNC C2	178.208.87[.]21
DanaBot C2	77.91.73[.][187
DanaBot C2	74.119.193[.]200

Reference

eSentire Threat Response Unit (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.