securelist.com /hrserv-apt-web-shell/111119/

HrServ – Previously unknown web shell used in APT attack

Authors

- Mert Degirmenci

Introduction

In the course of our routine investigation, we discovered a DLL file, identified as hrserv.dll, which is a previously unknown web shell exhibiting sophisticated features such as custom encoding methods for client communication and in-memory execution. Our analysis of the sample led to the discovery of related variants compiled in 2021, indicating a potential correlation between these separate occurrences of malicious activity.

Initial infection

According to our telemetry data, the PAExec.exe process initiates the creation of a scheduled task on the system named MicrosoftsUpdate (sic), which in turn is designed to execute a .BAT file.

```
"schtasks" /create /sc DAILY /tn MicrosoftsUpdate /tr "$system32\cmd.exe /c
$public\JKNLA.bat $public\hrserv.dll" /ru system /f
```

The .BAT file accepts the path of a DLL file as an argument. In this instance, the script is provided with the file $public\hrserv.dll, which is then copied to the System32 directory. After this operation, the script configures a service via the system registry and the sc utility. It then activates the newly created service.

HrServ web shell

MD5	418657bf50ee32acc633b95bac4943c6
SHA1	cb257e00a1082fc79debf9d1cb469bd250d8e026
SHA256	8043e6c6b5e9e316950ddb7060883de119e54f226ab7a320b743be99b9c10ec5
Link time	2023-Aug-30 08:28:15
File type	PE32+ executable (DLL) (console) x86-64, for MS Windows
Compiler	Microsoft Visual C/C++(2015 v.14.0)

The sequence of operations starts with the registration of a service handler. HrServ then initiates an HTTP server utilizing the HTTP server API for its functionality. It calls the HttpAddUrlToGroup function to register the following URL so that matching requests are routed to the request queue.

```
http://+:80/FC4B97EB-2965-4A3B-8BAD-B8172DE25520/
```

Client-server communication uses custom encoding techniques that include Base64 encoding and FNV1A64 hashing algorithms.

Based on the type and information within an HTTP request, specific functions are activated. These functions are distinguished by the GET parameter named cp. In addition, the DLL file utilizes the value of the NID cookie for various purposes. The use of the GET parameter pattern and the cookie value is consistent with practices employed by Google. We suspect that this intentional similarity in naming conventions is intended to disguise these requests in network traffic, making it more challenging to detect such malicious activity.

An example of such a request would be:

```
&cp=1&client=desktop-gws-wiz-on-focus-serp&xssi=t&hl=en-TW&authuser=0&pq=
```

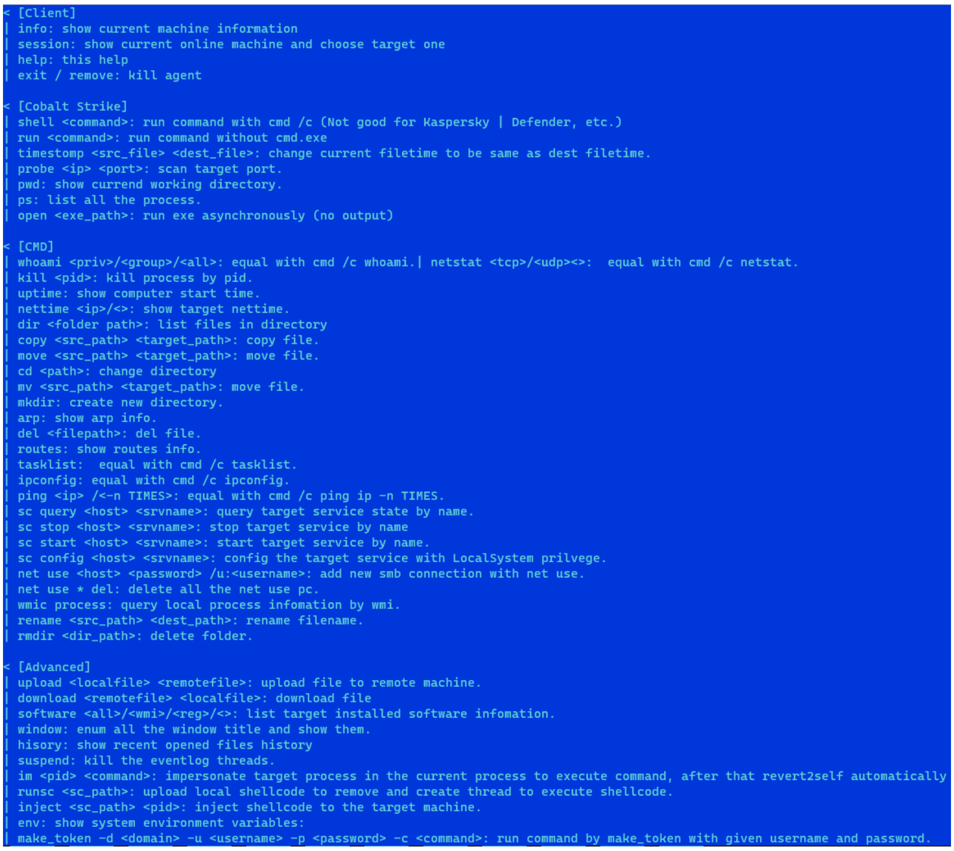
Request type	cp value	Description
GET	0	Call VirtualAlloc and copy a custom decoded NID cookie value, then create a new thread.
POST	1	Create a file using the custom decoded NID cookie value and write the custom decoded POST data to that file.
GET	2	Read a file using the custom decoded NID cookie value and return it as a response by appending it to the end of the "data:image/png;base64" string; If an error occurs while reading the file, HrServ responds with the string: data:image/png;base64,c3Dlc+DheRzlKBV2Yh92KS//;
GET	4	Return Outlook Web App HTML data.
POST	6	Call VirtualAlloc and copy the custom decoded POST data, then create a new thread.
GET	7	Return Outlook Web App HTML data *[Duplicate]*.

Code execution

If the cp value in the request is 6, this indicates a code execution process.

- Initially, it extracts the value of the NID cookie and applies its custom decoding technique
- It writes this decoded value to the specified registry path, denoted as "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\IdentityStore\RemoteFile"
- The custom-decoded POST data is then copied to the memory, after which a new thread is created and the process enters a sleep state.

In a particular observed scenario, the cp value is unknown. A multifunctional implant is activated in the system memory. The implant creates a file in the directory "%temp%", retrieves information from the registry, performs some actions based on this information, and records the output of these actions in the created file. As a result, the registry and the temporary file are used as a communication channel between the implant and HrServ.



```
< [Client]
| info: show current machine information
| session: show current online machine and choose target one
| help: this help
| exit / remove: kill agent

< [Cobalt Strike]
| shell <command>: run command with cmd /c (Not good for Kaspersky | Defender, etc.)
| run <command>: run command without cmd.exe
| timestomp <src_file> <dest_file>: change current filetime to be same as dest filetime.
| probe <ip> <port>: scan target port.
| pwd: show currend working directory.
| ps: list all the process.
| open <exe_path>: run exe asynchronously (no output)

< [CMD]
| whoami <priv>/<group>/<all>: equal with cmd /c whoami.| netstat <tcp>/<udp><>:  equal with cmd /c netstat.
| kill <pid>: kill process by pid.
| uptime: show computer start time.
| nettime <ip>/<>: show target nettime.
| dir <folder path>: list files in directory
| copy <src_path> <target_path>: copy file.
| move <src_path> <target_path>: move file.
| cd <path>: change directory
| mv <src_path> <target_path>: move file.
| mkdir: create new directory.
| arp: show arp info.
| del <filepath>: del file.
| routes: show routes info.
| tasklist:  equal with cmd /c tasklist.
| ipconfig: equal with cmd /c ipconfig.
| ping <ip> /<-n TIMES>: equal with cmd /c ping ip -n TIMES.
| sc query <host> <srvname>: query target service state by name.
| sc stop <host> <srvname>: stop target service by name
| sc start <host> <srvname>: start target service by name.
| sc config <host> <srvname>: config the target service with LocalSystem prilvege.
| net use <host> <password> /u:<username>: add new smb connection with net use.
| net use * del: delete all the net use pc.
| wmic process: query local process infomation by wmi.
| rename <src_path> <dest_path>: rename filename.
| rmdir <dir_path>: delete folder.

< [Advanced]
| upload <localfile> <remotefile>: upload file to remote machine.
| download <remotefile> <localfile>: download file
| software <all>/<wmi>/<reg>/<>: list target installed software infomation.
| window: enum all the window title and show them.
| hisory: show recent opened files history
| suspend: kill the eventlog threads.
| im <pid> <command>: impersonate target process in the current process to execute command, after that revert2self automatically
| runsc <sc_path>: upload local shellcode to remove and create thread to execute shellcode.
| inject <sc_path> <pid>: inject shellcode to the target machine.
| env: show system environment variables:
| make_token -d <domain> -u <username> -p <password> -c <command>: run command by make_token with given username and password.
```

Available commands of the memory implant

Based on our telemetry data, after successfully establishing a foothold and placing the memory implant in the system memory, the next actions are to erase the previously existing traces by deleting the scheduled "MicrosoftsUpdate" job and both the initial DLL and batch files:

```
schtasks /delete /tn MicrosoftsUpdate /f
cmd /c "del /f/s/q $public\hrserv.dll & del /f/s/q $public\JKNLA.bat"
```

Older variants

We have also discovered earlier, differently named variants of HrServ. These DLL files date back to early 2021. They also use the custom encoding algorithm and behave the same way after a file read error. However, there are subtle differences.

- The web shell URL of these older variants differs from the current one:

  ```
  https://+:443/owa/MSExchangeService.svc
  ```
- These samples exhibit a distinct behavior by creating a process and retrieving its output through a pipe, as opposed to allocating a memory section and creating a thread from it.

Victims

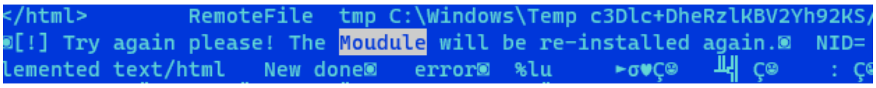
The only known victim according to our telemetry is a government entity in Afghanistan.

Attribution

The TTPs analyzed in this investigation are not associated with any known threat actors we are tracking, but there are a few things that we observed:

- the GET parameters used in the hrserv.dll file, which is used to mimic Google services, include “hl”. This specifies the host language of the user interface. Although this parameter has no functionality within the attack vector, the assigned value “en-TW” specifies that the Google search interface should be displayed in English, but the search results should be displayed in Traditional Chinese:

  ```
  &cp=1&client=desktop-gws-wiz-on-focus-serp&xssi=t&hl=en-TW&authuser=0&pq=
  ```
- the samples include help strings for specific conditions, in English. We saw multiple typos that suggest the actor behind the samples is not a native English speaker.



```
</html>         RemoteFile  tmp C:\Windows\Temp c3Dlc+DheRzlKBV2Yh92KS/
◙[!] Try again please! The Moudule will be re-installed again.◙  NID=
lemented text/html   New done◙   error◙  %lu      ►σ♥Ç☻   ╜╢ Ç☻    : Ç☻
```

An error message with a typo

Conclusion

The analyzed sample represents a capable web shell. Based on the compile timestamps, its origins date back to at least 2021. This sophisticated malware variant exhibits the ability to initiate in-memory executions. In the observed scenario, communication is established through registry manipulations and temporary files.

Notably, the web shell and memory implant use different strings for specific conditions. In addition, the memory implant features a meticulously crafted help message. Considering these factors, the malware’s characteristics are more consistent with financially motivated malicious activity. However, its operational methodology exhibits similarities with APT behavior. Despite the malware’s prolonged activity over several years, multiple instances involving these samples have not been documented. Our efforts are ongoing as we continue to monitor related activity, with the goal of unraveling the mystery in future investigations.

Indicators of compromise

File hashes

b9b7f16ed28140c5fcfab026078f4e2e
418657bf50ee32acc633b95bac4943c6
d0fe27865ab271963e27973e81b77bae
890fe3f9c7009c23329f9a284ec2a61b