You will always remember this as the day you finally caught FamousSparrow

By Alexandre Côté Cyr

Archived: 2026-04-05 20:37:39 UTC

In July 2024, ESET Research noticed suspicious activity on the system of a trade group in the United States that operates in the financial sector. While helping the affected entity remediate the compromise, we made an unexpected discovery in the victim's network: malicious tools belonging to FamousSparrow, a China-aligned APT group. There had been no publicly documented FamousSparrow activity since 2022, so the group was thought to be inactive. Not only was FamousSparrow still active during this period, it must have also been hard at work developing its toolset, since the compromised network revealed not one, but two previously undocumented versions of SparrowDoor, FamousSparrow's flagship backdoor.

Both of these versions of SparrowDoor constitute marked progress over earlier ones, especially in terms of code quality and architecture. One of them resembles the backdoor that researchers at Trend Micro called CrowDoor and attributed to the Earth Estries APT group in November 2024. The other is modular and significantly different from all previous versions. This campaign is also the first documented time FamousSparrow used ShadowPad, a privately sold backdoor, known to only be supplied to China-aligned threat actors.

We further discovered that, as part of this campaign, the threat actor managed to breach a research institute in Mexico just a couple of days prior to the compromise in the US.

While setting up tracking based on what we discovered in these attacks, we uncovered additional activity by the group between 2022 and 2024, which we're still investigating. Among others, it targeted a governmental institution in Honduras.

This blogpost provides an overview of the toolset used in the July 2024 campaign, focusing on the undocumented versions of the SparrowDoor backdoor that we discovered at the US victim.

> **Key points of this blogpost:**
>
> - ESET researchers discovered that FamousSparrow compromised a trade group for the financial sector in the United States and a research institute in Mexico.
> - FamousSparrow deployed two previously undocumented versions of the SparrowDoor backdoor, one of them modular.
> - Both versions constitute considerable progress over previous ones and implement parallelization of commands.
> - The APT group was also observed using the ShadowPad backdoor for the first time.
> - We discuss Microsoft Threat Intelligence's attribution claims linking FamousSparrow to Salt Typhoon.

FamousSparrow is a cyberespionage group with ties to China, active since at least 2019. We first publicly documented the group in a 2021 blogpost when we observed it exploiting the ProxyLogon vulnerability. The group was initially known for targeting hotels around the world, but has also targeted governments, international organizations, engineering companies, and law firms. FamousSparrow is the only known user of the SparrowDoor backdoor.

Even though FamousSparrow seemed inactive at the time of our discovery, we attribute this activity to the group with high confidence. The deployed payloads are new versions of SparrowDoor, a backdoor that appears to be exclusive to this group. While these new versions exhibit significant upgrades in code quality and architecture, they can still be traced back directly to earlier, publicly documented versions. The loaders used in these attacks also present substantial code overlaps with samples previously attributed to FamousSparrow. Notably, they use the same reflective loader shellcode as the libhost.dll loader sample described in a report from February 2022 published by the UK National Cyber Security Centre (NCSC). Its configuration also shares the same specific format, except for the encryption key which is instead hardcoded in the loader and backdoor. XOR encryption has also been replaced with RC4.

Additionally, C&C server communications use a format very similar to that used in previous SparrowDoor versions.

In 2021, Kaspersky researchers wrote about a threat actor they track as GhostEmperor. Despite some infrastructure overlap with FamousSparrow, we track them as separate groups. In August 2023, Trend Micro noted that some FamousSparrow TTPs overlap with those of Earth Estries. We have also observed code overlaps between SparrowDoor and that group's HemiGate. These are discussed in more detail in the *Plugins* section. We believe that the two groups overlap at least partially, but we do not have enough data to fully assess the nature and extent of the link between the two groups.

FamousSparrow and Salt Typhoon

Before we dive into the analysis of FamousSparrow's toolset, we want to discuss our position on the links between FamousSparrow and Salt Typhoon made by Microsoft Threat Intelligence.

In September 2024, the Wall Street Journal published an article (the article is behind a paywall) reporting that internet service providers in the United States had been compromised by a threat actor named Salt Typhoon. The article relays claims by Microsoft that this threat actor is the same as FamousSparrow and GhostEmperor. It is the first public report that conflates the latter two groups. However, as we already stated, we see GhostEmperor and FamousSparrow as two distinct groups. There are few overlaps between the two but many discrepancies. Both used 27.102.113[.]240 as a download server in 2021. Both groups were also early exploiters of the ProxyLogon vulnerability (CVE-2021-26855) and have used some of the same publicly available tools. However, besides these publicly available tools, each threat actor has its own custom toolset.

Since that initial publication, researchers at Trend Micro have added Earth Estries to the list of groups that are linked to Salt Typhoon. As of this writing, Microsoft, who created the Salt Typhoon cluster, has not published any technical indicators or details about TTPs used by the threat actor, nor provided an explanation for this attribution.

To avoid further muddying the waters, we will keep tracking the cluster of activity we see as directly linked to SparrowDoor as FamousSparrow until we have information necessary to reliably assess these attribution claims.

Based on our data and analysis of the publicly available reports, FamousSparrow appears to be its own distinct cluster with loose links to the others mentioned in this section. We believe those links are better explained by positing the existence of a shared third party, such as a digital quartermaster, than by conflating all of these disparate clusters of activity into one.

Technical Analysis

In order to gain initial access to the affected network, FamousSparrow deployed a webshell on an IIS server. While we were unable to determine the exact exploit used to deploy the webshells, both victims were running outdated versions of Windows Server and Microsoft Exchange, for which there are several publicly available exploits.

As for the toolset used in the campaign, the threat actor employed a mix of custom tools and malware along with those shared by China-aligned APT groups, as well as from publicly available sources. The final payloads were SparrowDoor and ShadowPad. Figure 1 provides an overview of the compromise chain deployed in the attacks.

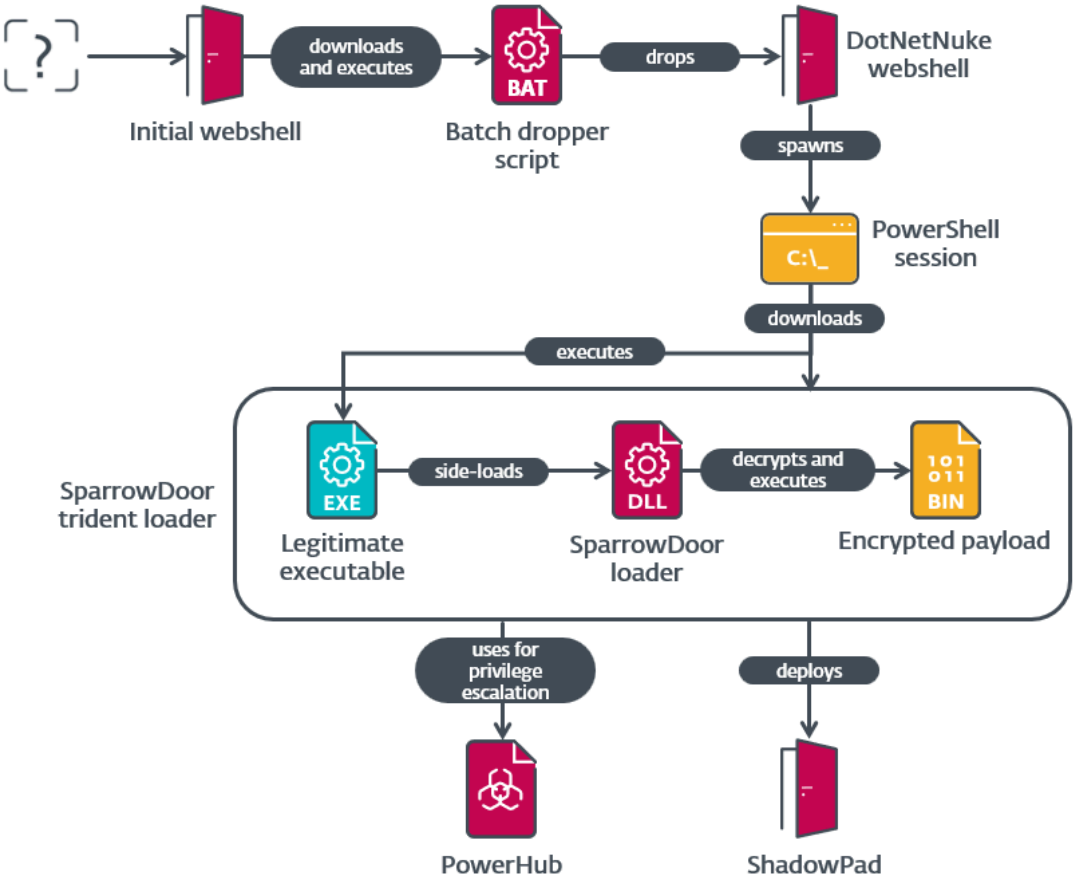

Figure 1. Overview of the compromise chain used in this FamousSparrow campaign

The threat actor initially downloaded a batch script over HTTP from a download server, 43.254.216[.]195. This script contains a base64-encoded .NET webshell that it writes to C:\users\public\s.txt. It then decodes it using

certutil.exe and saves the decoded output to C:\users\public\s.ashx. An ASHX module is a type of HTTP handler for ASP.NET. Although similar to ASPX modules, ASHX modules do not include any user interface components. The script then walks through drives C: to I:, and P:, to find the installation directory of DotNetNuke; it then copies the ASHX webshell to <DotNetNuke_directory>\DesktopModules\DotNetNuke.ashx.

The webshell itself is fairly generic and doesn't use anything specific to DotNetNuke. All the data it receives, and returns, is AES encrypted with the hardcoded key e2c99096bcecd1b5. On first request, it expects a .NET PE file. This executable file is loaded into memory and saved in a session variable. On subsequent requests, an instance of the LY class contained within that .NET assembly is created and the data received is passed to its Equals method. We did not collect any payload sent to this webshell, but it's obvious that the Equals method does not follow the typical contract.

In the cases we observed, this was used to spawn an interactive remote PowerShell session. Once this session was established, attackers used legitimate Windows tools to obtain information about the host and the Active Directory domains to which it was joined. They then downloaded PowerHub, an open-source post-exploitation framework, from an attacker-controlled server and used the BadPotato privilege-escalation technique to gain SYSTEM privileges. This exploit is not present in the framework, but it appears that the group added the open-source Invoke-BadPotato module to its deployment of PowerHub. Finally, the attacker used PowerShell's built-in Invoke-WebRequest to download three files from the same server that comprise SparrowDoor's trident loader.

In a process very similar to the one described in 2022 by the UK NCSC, the aforementioned files use a trident loading scheme to execute SparrowDoor. In this instance, the executable used for DLL side-loading is a legitimate version of K7AntiVirus Messenger Scanner named K7AVMScn.exe, while the malicious DLL and encrypted payload files are named K7AVWScn.dll and K7AVWScn.doc, respectively. The payload file is encrypted using an RC4 key that is hardcoded in both the loader and the resulting decrypted payload, but which varies across samples.

The decrypted payload consists of a custom configuration and reflective loader shellcode almost identical to that described by the UK NCSC, with the only difference being that the first field, which contained the four-byte XOR key, has been removed. The last 202 bytes of the file are encrypted separately, but using the same RC4 key, and contain the C&C server configuration.

SparrowDoor

As stated, we observed two new versions of SparrowDoor used in these attacks. The first one is very similar to what was called CrowDoor by researchers at Trend Micro, in an article published in November 2024 about Earth Estries. This malware was first documented by researchers at ITOCHU and Macnica in a presentation at VirusBulletin in 2023. From our perspective, these are part of the continued development effort on SparrowDoor rather than a different family. We can follow the evolution from the first version we described in 2021, through the ones referred to as CrowDoor, to the modular version we analyze in the later part of this blogpost.

Both versions of SparrowDoor used in this campaign constitute onsiderable advances in code quality and architecture compared to older ones. The most significant change is the parallelization of time-consuming commands, such as file I/O and the interactive shell. This allows the backdoor to continue handling new

commands while those tasks are performed. We will explain the procedure later in the blogpost when we discuss the commands in detail.

Just like in previous versions, the behavior of the backdoor varies depending on the command line argument passed to it. These are listed in Table 1.

Table 1. Command line arguments for SparrowDoor

Argument	**Behavior**
No argument	Establish persistence.
11	Process hollowing of colorcpl.exe.
22	Main backdoor operation.

When executed without any arguments, the malware establishes persistence. It first tries to do so by creating a service named K7Soft that is set to run automatically on startup. If this fails, a registry Run key with the same name is used instead. In both cases, the persistence mechanism is set to execute the backdoor with a command line argument of 11. It is also launched immediately with that same argument using the StartServiceA or ShellExecuteA API.

When executed with the argument 11, the backdoor launches the Windows color management tool (colorcpl.exe) with a command line argument of 22 and injects its loader into the newly created process.

It is only when the command line argument is set to 22 that the backdoor actually executes its main payload.

After SparrowDoor is executed in this backdoor mode, it terminates, in a roundabout way, any other already running instances. The backdoor uses the K32EnumProcesses API to iterate through the process IDs (PIDs) of all running processes and tries to create a mutex named Global\ID(<PID>). PIDs of 15 or less are skipped, likely as a way to exclude killing some essential system processes. If the mutex already exists, the process is terminated. Otherwise, the mutex is closed immediately. When SparrowDoor is done iterating through the PIDs, it creates a new mutex using the same name format and its own PID.

The backdoor then reads the last 202 bytes from the encrypted payload file and decrypts them using the same RC4 key used by the loader. The resulting plaintext is the C&C server configuration, which consists of three pairs of addresses and ports, followed by four numeric values that, respectively, represent the number of days, hours, minutes, and seconds the backdoor should wait after all configured C&C servers have been tried. This is related to the functionality we describe later while talking about the command the backdoor uses for changing the C&C configuration.

After loading this configuration, the backdoor will try to connect to the first server. If it is unable to connect or if the C&C server issues a command that causes execution to exit the main command loop, SparrowDoor will try to connect to the next server, and so on. Once the last server in the configuration has been tried, the backdoor will sleep for the defined time (six minutes in the sample we analyzed), reload the configuration, and then repeat the process. Note that, during this time, SparrowDoor does not respond to commands. However, the parallelized

commands that were already running will keep doing so until they complete, encounter an error, or are terminated by the server.

The backdoor uses two classes to manage its connections: the abstract CBaseSocket and its child class CTcpSocket. These are essentially wrappers around Winsock TCP sockets. While the class names are generic and follow the same naming convention used in the Microsoft Foundation Class Library (MFC), the code they contain appears to be custom.

SparrowDoor uses an integer value as a victim or session identifier. This is sent to the C&C server when it requests information about the host and whenever a new socket is created. The value is read from the HKLM\Software\CLASSES\CLSID\ID registry key, falling back to the same path in the HKCU hive if there's an issue. If it is not present, the identifier is derived from the machine's performance counter and written to the aforementioned registry key. Although the value itself is benign, the use of this nonstandard registry key presents a detection opportunity. Indeed, the name of any registry key under Software\Classes\CLSID\ should be a valid CLSID, which are represented as a GUID surrounded by curly brackets. While it is not necessarily an indicator of maliciousness, the presence of keys with nonstandard names under CLSID is unusual.

Commands

The first version of SparrowDoor used in this campaign supports more commands, described in Table 2, than previously documented versions. While the command IDs are different from those used in the version analyzed by Trend Micro, the order and offset between IDs are the same. We have not had access to that sample, so we cannot tell whether the additional commands were absent or simply not publicly documented by the authors.

As previously mentioned, some of the commands have been parallelized. When the backdoor receives one of these commands, it creates a thread that initiates a new connection to the C&C server. The unique victim ID is then sent over the new connection along with a command ID indicating the command that led to this new connection. This allows the C&C server to keep track of which connections are related to the same victim and what their purposes are. Each of these threads can then handle a specific set of subcommands. To limit its complexity, Table 2 does not include these subcommands; we will go over them separately.

Table 2. Main commands implemented by SparrowDoor

Command ID	Description	Received data	Sent data
0x32341122	Initial connection.	*No message*	*Empty*
0x32341123	Send host information.	*Empty*	· IP address, · unique ID, · OS build number, · OS major version number, · OS minor version number, · computer name, and · username.

Command ID	Description	Received data	Sent data
0x32341124	Start interactive shell session (parallel).	*Empty*	See the *Interactive shell* subsection.
0x32341127	Sleep, then move to the next server in the configuration.	Minutes to sleep.	*No response*
0x32341128	Uninstall backdoor and clean up.	*Empty*	*No response*
0x32341129	Get current network configuration.	*Empty*	Network configuration structure.
0x3234112A	Set network configuration.	Network configuration structure.	*No response*
0x3234112B	Execute loader with the command line argument 11 and terminate the current process.	*Empty*	*No response*
0x3234112D	File I/O (parallel).	Operation ID.	See the File operations section.
0x32341131	Get information about connected drives.	*Empty*	Array of 26 bytes representing the drive type of all drives from A: to Z: as returned by GetDriveTypeW.
0x32341132	List files.	Directory path.	File information, one response per file. See the *File listing* section.
0x32341135	Create directory.	Directory path.	*No response*
0x32341136	Move or rename file.	· Source path length, · source path, · destination path length, and · destination path.	*No response*
0x32341137	Delete file.	File path.	*No response*
0x32341138	Start proxy.	*Empty*	See the *Proxy* subsection.

All communication between the malware and its C&C server uses the same base packet format, defined in Figure 2. The format of the data section depends on the command sent, and can be empty. In most cases, responses use the ID of the command to which the backdoor is responding. There are, however, some exceptions; we will describe these when talking about the relevant commands in detail.

```
struct Command {
    DWORD NULL;
    unsigned int command_id;
    unsigned int data_length;
    DWORD NULL;
    BYTE data[data_length];
}
```

Figure 2. Base packet format used for network communication

Interactive shell

Upon receiving the interactive shell command, SparrowDoor spawns a new thread and socket as previously described, and performs all the following actions within this thread using the new socket. First, the backdoor sends back an acknowledgment message with command ID 0x32341125 and the unique victim ID in the data field. It then spawns a cmd.exe process and uses a pair of threads and named pipes to relay commands and their output between the C&C server and the shell. The named pipe \\.\pipe\id2<address> is used to pass commands received from the C&C server to the shell and \\.\pipe\id1<address> is used for the resulting output on STDOUT and STDERR. In both instances, <address> is the memory address, in decimal form, of the CTcpSocket instance. These commands use the ID 0x32341126 and the data is, respectively, the command line to be executed and the raw output. If the backdoor receives a message with the command ID set to any other value, the interactive shell session is terminated.

Changing the C&C configuration

The C&C configuration is kept in the encrypted payload file. If the backdoor receives the command to change this configuration (0x3234112A), the received structure is RC4 encrypted and then the last 202 bytes of the encrypted file are overwritten with the result. Interestingly, the configuration is not automatically reloaded. As we explained previously, the configuration is only reloaded when all three configured servers have been tried. To forcibly reload the configuration, the server can issue the 0x32341127 command or an invalid command, both of which will cause SparrowDoor to exit the command loop and move to the next server. The configuration is also reloaded if the backdoor is relaunched, such as by using the 0x3234112B command.

File operations

As with other commands processed in parallel, everything here is performed in a new thread using a new socket. SparrowDoor sends an acknowledgment message with the same ID as the original command. The body of this

message contains the unique ID of the victim and the operation ID sent by the C&C server. This operation ID does not appear to have any meaning, and is probably only used by the server to link the connection to the file operation command if multiple such commands are performed in parallel. Command IDs 0x3234112E and 0x3234112F are used, respectively, for file reads and writes.

For a file read, the message body contains the starting offset, the size to be read, and the path to the file. If the requested read goes past the end of the file, it causes an error and no response is sent. Otherwise, the malware reads the file in chunks of 4 kB, each of which is sent in the body of a message with the command ID 0x32341130.

The process is similar for a file write. The initial message from the C&C contains the total size of the data to be written followed by the target file path. Interestingly, the write is only performed if this size is greater than the current size of the target file. The data is then sent by the C&C server in chunks of 4 kB, using the same command ID of 0x32341130.

File listing

When the file listing command is received, the backdoor first sends back an acknowledgment message with the command ID 0x32341133. It then uses the FindFirstFileW and FindNextFileW API functions to iterate, non-recursively, through files in the target directory. For each file, SparrowDoor sends one message, with the same command ID as the list file command (0x32341132) and the information described in Figure 3. Note that, even though the length of the filename isn't specified directly, it can be obtained by subtracting the size of the rest of the fields (0x16) from the data_length value in the header.

```
struct FileInformation {
    DWORD FileAttributes;
    DWORD FileSizeHigh;
    DWORD FileSizeLow;
    DWORD LastWriteTimeLow;
    DWORD LastWriteTimeHigh;
    WORD offset_to_filename // hardcoded to 0x16
    char filename[data_length - 0x16]
}
```

Figure 3. Format of the information sent for each listed file

Once the iteration is done, a message with command ID 0x32341134 and no data is sent to indicate that the file listing operation has completed successfully.

Proxy

This functionality allows the backdoor to act as a TCP proxy between the C&C server and an arbitrary machine. As with other commands processed in parallel, the following is done in a new thread using its own socket. SparrowDoor sends an acknowledgment message with the same ID as the original command; the body of this message contains the unique ID of the victim. Command ID 0x32341139 is then sent by the server to actually initiate the proxy. The proxy functionality is achieved by creating two new sockets, one connected to the C&C server and another connected to an address and port provided by the server on that new connection. SparrowDoor then uses a pair of Winsock structures and events to keep track of incoming packets and relay them between the two parties. The addition of proxy functionality to SparrowDoor may be a hint that the group is following the trend of China-aligned threat actors building and using operational relay box (ORB) networks.

Modular SparrowDoor

The modular version of SparrowDoor is significantly different from the previous ones. On the network communication side, the command header is sent separately from the body and that data is RC4 encrypted with the hardcoded key iotrh^%4CFGTj. The custom classes used for network communication in this version still use Winsock TCP sockets and are very similar to those we mentioned previously – the most notable difference being that the child class is deceptively named CShttps instead of CTcpSocket. As seen in Table 3, of the commands present in previous versions of SparrowDoor, this one only implements the commands that relate to managing the C&C configuration and uninstalling the backdoor. Information about the host machine is sent automatically after the initial connection message and includes a list of installed security products in addition to what was sent in previous versions.

All of the other commands are related to the handling of plugins. We believe that the removed functionality has simply been moved to one or more modules. While we have yet to observe any such plugin, we can share insights based on our analysis of the code that implements this functionality.

Table 3. Commands implemented in the modular version of SparrowDoor

Command ID	Response ID	Description
N/A	0x136433	Initial connection.
N/A	0x0A4211	Send host information.
0x3A72	0x0A4214	Get current network configuration.
0x3A73	*No response*	Set network configuration.
0x3A75	0x136434	Initiate plugin command loop. See the *Plugins* subsection.
0x3A76	0x136435 / 0x0A4217	
0x3A77	0x136435 / 0x0A421F	
0x3A78	0x136435 / 0x0A4221	
0x3A7B	0x136435 / 0x0A4228	

Command ID	Response ID	Description
0x3A7A	*No response*	Uninstall backdoor and clean up.

Plugins

Installed plugins are referenced via a standard C++ list; each entry consists of a bitmask and a handler function address. The bitmask is used to determine which command IDs are handled by the plugin and corresponds to the low nibble of the third byte of the command ID (i.e., CommandID & 0xF0000).

This version of SparrowDoor can use five different command IDs to invoke plugin commands. Of those, three (0x3A76, 0x3A77, and 0x3A7B) follow almost exactly the same path in the code – the only difference being the response ID of the acknowledgment message. There are some very minor differences in the handshake process between this set of commands and the other two. However, in all cases, the command is parallelized using the same method we described in the *Commands* section. On the new socket, the backdoor sends the corresponding response ID, the unique host ID, and the data it initially received from the C&C server. This data appears to function like the operation ID mentioned in the *File operations* section. After this handshake is completed, all five commands call the same function to actually handle the plugin command. This function receives the command ID and data from the C&C server, then iterates through installed plugins to dispatch the command to the correct handler. The process is repeated until the backdoor receives an incorrectly formatted command message.

By default, only one plugin, with a bitmask of 0x10000, is installed. This plugin handles the installation of new plugins sent by the C&C server. Plugins are sent by the server as PE files and are never stored on disk. Coupled with the reduced function set present in the base backdoor, this is probably meant to evade detection. After such a plugin is received, it is manually mapped in memory and its fmain export is called. This function returns a pointer to a structure containing the address of a function that returns the bitmask for the plugin and the address of the handler function. If no installed plugin has the same bitmask, the newly received plugin is added to the list.

Links to previous versions

We have also identified older samples that present significant code overlaps with this modular version, including similar code to handle plugins. These samples correspond to the backdoor that Trend Micro named HemiGate in an August 2023 article. Some of the samples even use the same RC4 key mentioned in that article. Rather than being sent by the C&C, plugins are implemented as C++ classes inheriting from an abstract class named PluginInterface. These plugins follow the same pattern described in the previous paragraph: they have a method that returns a bitmask, used to dispatch commands, and a second method to handle commands. We believe that HemiGate represents an earlier step in the evolution of the modular backdoor. Thus, it is likely that the plugins contained therein are representative of those used in the more recent modular version. Table 4 presents an overview of the plugins and their functionality.

Table 4. Summary of plugins contained in HemiGate

Bitmask	Class name	Description
0x20000	Cmd	Run a single command.
0x30000	CFile	File system operations.
0x40000	CKeylogPlug	Keylogger functionality.
0x50000	CSocket5	TCP proxy. This is very similar to the functionality described earlier in the *Proxy* section.
0x60000	CShell	Interactive shell.
0x70000	CTransf	File transfer between the client and C&C server.
0x80000	CRdp	Take screenshots.
0xA0000	CPro	· List running processes. · Kill a process.
0xC0000	CFileMoniter	Monitor file system changes for specified directories.

These similarities are evidence that the cluster we track as FamousSparrow at least partially overlaps with Earth Estries. Since HemiGate pre-dates both versions of SparrowDoor detailed earlier in this report, it may also be an indication that the modular and the parallelized versions of SparrowDoor are being developed in parallel.

ShadowPad

After SparrowDoor was detected in the US victim's network, it was used to execute an MFC-based loader bearing similarities to the ShadowPad loaders previously documented by Cisco Talos.

This ShadowPad loader is a DLL named imjpp14.dll, meant to be loaded via DLL side-loading by the more-than-14-year-old, legitimate, outdated version of the Microsoft Office IME executable, imecmnt.exe, renamed to imjp14k.exe. The loader first checks whether its current process is the expected side-loading host by performing pattern matching at offset 0xE367 in-memory. Once this validation succeeds, the malicious DLL decrypts the file named imjp14k.dll.dat that is located in the same directory as the DLL and its side-loading host. Finally, the decrypted payload is injected into a wmplayer.exe process (Windows Media Player).

Even though we did not retrieve the encrypted payload, an in-memory ShadowPad detection occurred in a wmplayer.exe process, with impjp14k.exe as its parent process. Furthermore, it connected to a ShadowPad C&C server (IP: 216.238.106[.]150). While we didn't observe any ShadowPad sample using it, one of the SparrowDoor C&C servers had a TLS certificate matching a known ShadowPad fingerprint.

Additionally, we detected ShadowPad loaders and the ShadowPad backdoor in memory on several machines in the victim's network.

Note that this is the first time we have observed FamousSparrow making use of the ShadowPad backdoor.

Other tools

During the compromise, in addition to the various malware mentioned above, we also observed the following being used by the threat actor:

- A basic batch script that dumps the registry with the following commands:
  - reg save HKLM\SYSTEM C:\users\public\sys.hiv,
  - reg save HKLM\SAM C:\users\public\sam.hiv, and
  - reg save hklm\security C:\users\public\security.hiv.
- Impacket or NetExec, detected by our firewall, but we have not collected any of the commands.
- A loader for a version of the open-source Spark RAT that was modified to include code from an open-source Go shellcode loader.

We also noticed the use of a tool to dump LSASS memory with the undocumented MiniDumpW API function. This tool is split into two DLLs stored on disk as %HOME%\dph.dll and %WINDIR%\SysWOW64\msvc.dll. The latter is probably meant to blend in with the legitimate libraries for Microsoft Visual C++ (MSVC) that are stored in the same directory. The former is loaded via a legitimate version of VLC's Cache Generator (vlc-gen-cache.exe), renamed to dph.exe, and imports functions from the latter. Since VLC plugins can be native DLLs, its cache generator naturally contains code to load and execute such libraries.

Network infrastructure

The ShadowPad C&C server uses a self-signed TLS certificate, with a SHA-1 fingerprint of BAED2895C80EB6E827A6D47C3DD7B8EFB61ED70B, that attempts to spoof those used by Dell. This follows the format that was described by Hunt Intelligence in an article from February 2024. While this pattern can be used to track ShadowPad servers, it is not linked to any specific threat actor. One of the C&C servers used by SparrowDoor (45.131.179[.]24:80) had a TLS certificate, on port 443, with the same Common Name (CN) as the certificate used by the aforementioned ShadowPad C&C server. This server is also the only one that was present in both versions of SparrowDoor.

We observed three unique SparrowDoor C&C servers in this campaign, all of which used port 80. The modular sample was configured with amelicen[.]com as its third C&C server. When the sample was first detected, this domain pointed to the IP address mentioned in the previous paragraph. One of the C&C servers configured in the modular sample (43.254.216[.]195:80) was also used by the SparrowDoor loader. This is strange, since SparrowDoor uses plain TCP and the files were downloaded over HTTP. However, there is a gap of almost two weeks between the downloads, on June 30, 2024, and the compilation of the modular SparrowDoor, on July 12, 2024. We do not know whether the service listening on that port was changed between those two occurrences or whether the SparrowDoor C&C server includes functionality to serve files over HTTP.

Conclusion

Due to the lack of activity and public reporting between 2022 and 2024, FamousSparrow was presumed to be inactive. However, our analysis of the US network compromised in July 2024 revealed two new versions of SparrowDoor, showing that FamousSparrow is still developing its flagship backdoor. One of these new versions

was also found on a machine in Mexico. As we were setting up tracking based on what is covered in this blogpost, we uncovered additional activity by the group during this period, including the targeting of a governmental institution in Honduras. This newly found activity indicates that not only is the group still operating, but it was also actively developing new versions of SparrowDoor during this time.

We will continue to monitor and report on activity by FamousSparrow, and will also continue to follow the discussion surrounding potential links between FamousSparrow and Salt Typhoon.

> *For any inquiries about our research published on WeLiveSecurity, please contact us at threatintel@eset.com.*
>
> *ESET Research offers private APT intelligence reports and data feeds. For any inquiries about this service, visit the ESET Threat Intelligence page.*

IoCs

Files

SHA-1	Filename	Detection	Description
C26F04790C6FB7950D89AB1B08207ACE01EFB536	DotNetNuke.ashx	ASP/Webshell.SE	ASHX webshell.
F35CE62ABEEDFB8C6A38CEAC50A250F48C41E65E	DrmUpdate.exe	N/A	Legitimate Microsoft Office IME 2010 used for DLL side-loading.
5265E8EDC9B5F7DD00FC772522511B8F3BE217E3	imjp14k.dll	Win32/Agent.AGOZ	ShadowPad loader.
A91B42E5062FEF608F285002DEBAFF9358162B25	dph.exe	N/A	Legitimate VLC cache generator.
0DC20B2F11118D5C0CC46B082D7F5DC060276157	vlc.exe	N/A	Legitimate VLC media player used for DLL side-loading.
EF189737FB7D61B110B9293E8838526DCE920127	libvlc.dll	Win64/Agent.FAY	SparrowDoor loader.
D03FD329627A58B40E805F4F55B5D821063AC27F	notify.exe	N/A	Legitimate Yandex application used for DLL side-loading.
3A395DAAF518BE113FCFF2E5E48ACD9B9C0DE69D	WINMM.dll	Win32/Shellcode Runner.LK	Loader for modular SparrowDoor.

SHA-1	Filename	Detection	Description
0925F24082971F50EDD9 87D82F708845A6A9D7C9	WindowsUpdate.exe	N/A	Legitimate Fortemedia Audio Processing used for DLL side-loading.
5F1553F3AF9425EF5D68 341E991B6C5EC96A82EB	FmApp.dll	Win64/Agent.EEA	ShadowPad loader.
CC350BA25947B7F9EC5D 11EA8269407C0FD74095	FmApp.dll	Win64/Agent.EDQ	ShadowPad loader.
DB1591C6E23160A94F63 12CA46DA2D0BB243322C	K7AVWScn.exe	N/A	Legitimate K7AntiVirus Messenger Scanner Stub used for DLL side-loading.
1B06E877C2C12D74336E 7532BC0ECF761E5FA5D4	K7AVWScn.dll	Win32/Agent.AGOJ	SparrowDoor loader.
EBC93A546BCDF6CC1EB6 1D7174BCB85407BBD892	start.bat	BAT/Agent.DP	Batch script to deploy the ASHX webshell.
D6D32A1F17D48FE695C0 778018C0D51626DB4A3B	dph.dll	Win64/Riskware. LsassDumper.EN	Program to dump LSASS memory.
7D66B550EA68A86FCC09 58E7C159531D4431B788	Ntmssvc.dll	WinGo/Shellcode Runner.EC	Modified Spark RAT.
D78F353A70ADF68371BC 10CF869B761BD51484B0	N/A (in-memory)	Win32/Agent.VQI	Decrypted SparrowDoor payload.
99BED842B5E222411D19 F0C5B54478E8CC7AE68F	N/A (in-memory)	Win32/Agent.VQI	Decrypted modular SparrowDoor payload.
5DF3C882DB6BE1488718 2B7439B72A86BD28B83F	taskhosk.exe	Win32/Agent.AHCV	SparrowDoor/HemiGate with built-in plugins.
AA823148EEA6F43D8EB9 BF20412402A7739D91C2	taskhosk.exe	Win32/Agent.AHCV	SparrowDoor/HemiGate with built-in plugins.

Network

IP	Domain	Hosting provider	First seen	Details
43.254.216[.]195	N/A	Hongkong Wen Jing Network Limited	2024-06-27	FamousSparrow C&C and download server.

IP	Domain	Hosting provider	First seen	Details
45.131.179[.]24	amelicen[.]com	XNNET LLC	2024-07-05	SparrowDoor C&C server.
103.85.25[.]166	N/A	Starry Network Limited	2024-06-06	SparrowDoor C&C server.
216.238.106[.]150	N/A	Vultr Holdings, LLC	2024-03-11	ShadowPad C&C server.

MITRE ATT&CK techniques

This table was built using version 16 of the MITRE ATT&CK framework.

Tactic	ID	Name	Description
Resource Development	T1588.001	Obtain Capabilities: Malware	FamousSparrow acquired and used ShadowPad.
	T1588.002	Obtain Capabilities: Tool	FamousSparrow acquired the open-source PowerHub post-exploitation framework.
	T1588.005	Obtain Capabilities: Exploits	FamousSparrow added the BadPotato exploit to its deployment of PowerHub.
	T1583.004	Acquire Infrastructure: Server	FamousSparrow acquired a server to host malware and tools.
	T1584	Compromise Infrastructure	Servers compromised with SparrowDoor can be forced to function as proxies.
	T1608.001	Stage Capabilities: Upload Malware	FamousSparrow hosted SparrowDoor on its own server.
	T1608.002	Stage Capabilities: Upload Tool	FamousSparrow uploaded PowerHub to a server it controls.
	T1587.001	Develop Capabilities: Malware	FamousSparrow developed new versions of SparrowDoor.
Initial Access	T1190	Exploit Public-Facing Application	FamousSparrow likely exploited a vulnerability in an outdated Exchange server to gain initial access.
	T1078.002	Valid Accounts: Domain Accounts	FamousSparrow used valid credentials for a domain account to pivot to other

Tactic	ID	Name	Description
			machines in compromised networks.
Execution	T1059.001	Command-Line Interface: PowerShell	FamousSparrow used an interactive PowerShell session to perform reconnaissance and deploy SparrowDoor.
	T1059.003	Command-Line Interface: Windows Command Shell	SparrowDoor can launch cmd.exe to create a remote shell session.
	T1106	Native API	SparrowDoor uses the CreateProcess API to launch an interactive shell.
	T1047	Windows Management Instrumentation	FamousSparrow used wmic.exe to run reconnaissance commands.
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	SparrowDoor can create a Run key to persist on a compromised system.
	T1543.003	Create or Modify System Process: Windows Service	SparrowDoor can create a service to persist on a compromised system.
	T1505.003	Server Software Component: Web Shell	FamousSparrow deployed webshells to compromised servers.
Privilege Escalation	T1068	Exploitation for Privilege Escalation	FamousSparrow used the BadPotato exploit to gain SYSTEM privileges.
Defense Evasion	T1055	Process Injection	SparrowDoor injects its loader into a Windows color management process.
	T1055.001	Process Injection: Dynamic-link Library Injection	The ShadowPad loader injects its payload into a newly created Windows Media Player process.
	T1574.002	Hijack Execution Flow: DLL Side-Loading	The SparrowDoor loader is executed by side-loading from a legitimate K7 Antivirus executable.
	T1140	Deobfuscate/Decode Files or Information	SparrowDoor’s encrypted C&C server configuration is decrypted at runtime.
	T1564.001	Hide Artifacts: Hidden Files and Directories	FamousSparrow has used attrib.exe to set the hidden and system file attributes on the SparrowDoor loader.

Tactic	ID	Name	Description
	T1564.003	Hide Artifacts: Hidden Window	SparrowDoor launches the process into which it injects the loader, with its window hidden.
	T1070.004	Indicator Removal: File Deletion	SparrowDoor can uninstall itself, which includes deleting the associated files.
	T1070.009	Indicator Removal: Clear Persistence	SparrowDoor can uninstall itself, which removes any persistence.
	T1027.009	Obfuscated Files or Information: Embedded Payloads	FamousSparrow used a batch script that deploys an embedded ASPX webshell.
	T1027.010	Obfuscated Files or Information: Command Obfuscation	PowerHub obfuscates parts of its commands by encrypting them with RC4.
	T1027.013	Obfuscated Files or Information: Encrypted/Encoded File	The file containing the SparrowDoor payload is RC4 encrypted.
	T1036.004	Masquerading: Masquerade Task or Service	The description and name of the service used by SparrowDoor to persist match those of the legitimate K7 program it is impersonating.
	T1036.005	Masquerading: Match Legitimate Name or Location	The SparrowDoor loader masquerades as a DLL loaded by the legitimate K7AVWScn.exe.
	T1036.008	Masquerading: Masquerade File Type	The encrypted payload file containing SparrowDoor has a .doc extension.
	T1620	Reflective Code Loading	The modular version of SparrowDoor can load additional PE files into its own memory space.
Credential Access	T1003.001	OS Credential Dumping: LSASS Memory	FamousSparrow used a utility to dump LSASS memory.
Discovery	T1482	Domain Trust Discovery	FamousSparrow used nltest.exe to list domain controllers and trusted domains.

Tactic	ID	Name	Description
	T1087.001	Account Discovery: Local Account	FamousSparrow used net.exe to obtain information on local accounts.
	T1087.002	Account Discovery: Domain Account	FamousSparrow used net.exe to obtain information on domain accounts.
	T1049	System Network Connections Discovery	FamousSparrow used netstat.exe to list active TCP connections.
	T1083	File and Directory Discovery	SparrowDoor can list directories.
	T1057	Process Discovery	FamousSparrow used tasklist.exe to list running processes and services, and to find the LSASS process.
	T1012	Query Registry	FamousSparrow used a script to dump the SAM, SYSTEM, and SECURITY registry hives.
	T1082	System Information Discovery	FamousSparrow used wmic.exe to obtain information about mapped drives. It also used ipconfig.exe to list network adapters.
	T1033	System Owner/User Discovery	FamousSparrow used whoami.exe to obtain information about the active user and their privileges.
	T1518.001	Software Discovery: Security Software Discovery	The modular version of SparrowDoor lists installed security software.
Lateral Movement	T1570	Lateral Tool Transfer	FamousSparrow transferred SparrowDoor to other machines on the network.
	T1021	Remote Services	FamousSparrow has used remote PowerShell sessions to pivot onto other machines in the compromised network.
Collection	T1005	Data from Local System	SparrowDoor can read files from any local system drive.
	T1025	Data from Removable Media	SparrowDoor can read files from any mapped removable drive.

Tactic	ID	Name	Description
	T1039	Data from Network Shared Drive	SparrowDoor can read files from any mapped network shared drive.
Command and Control	T1095	Non-Application Layer Protocol	SparrowDoor uses raw TCP sockets to communicate with its C&C server.
	T1071.001	Application Layer Protocol: Web Protocols	FamousSparrow downloaded additional files from its C&C server over HTTP.
	T1573.001	Encrypted Channel: Symmetric Cryptography	In the modular version of SparrowDoor, data sent over the network is RC4 encrypted.
	T1008	Fallback Channels	SparrowDoor can have up to three C&C servers in its network configuration.
	T1105	Ingress Tool Transfer	FamousSparrow downloaded PowerHub from a server it controls.
	T1571	Non-Standard Port	FamousSparrow downloaded PowerHub over HTTP on port 8080 and over HTTPs on port 8443.
Exfiltration	T1020	Automated Exfiltration	SparrowDoor can exfiltrate the content of any file requested by the C&C server.
	T1030	Data Transfer Size Limits	SparrowDoor splits file content into chunks of 4 kB.
	T1041	Exfiltration Over C2 Channel	SparrowDoor exfiltrates data using the same raw TCP socket it uses to communicate with its C&C server.
