Detecting FIN7 attacks

FIN7 is a threat actor group which has primarily targeted the U.S. retail, restaurant, and hospitality sectors since mid-2015. FIN7 malware is commonly deployed through spear-phishing campaigns, using Windows 11 Word documents with Visual Basic macros to drop malicious payloads, including a JavaScript implant. The FIN7 group also uses REvil and Darkside ransomware payloads after gaining access to a compromised host.

You need to be able to detect and investigate unusual activities that might relate to FIN7 JS implant and its JSSLoader, including looking for image loading of LDAP and Windows Management Instrumentation (WMI) modules and other activities associated with FIN7's payload, data collection, and script execution.

Required data

- Endpoint data
- Windows Security Event logs

Some commands, parameters, and field names in the searches below might need to be adjusted to match your environment. In addition, to optimize the searches shown below, you should specify an index and a time range when appropriate.

How to use Splunk software for this use case

► Check elevated CMD using whoami

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. If you are using Sysmon, you must have at least version 6.0.4 of the Sysmon TA. You should also ensure you are ingesting normalized endpoint data, populating the Endpoint data model in the Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.

This search is designed to detect a suspicious whoami execution to check if command or shell instance processes are running with elevated privileges. This technique is used in FIN7 attacks as part of data collection on the infected machine.

This command is not commonly executed by a normal user or admin to check if a process is elevated. Tune and filter known instances where renamed rundll32.exe might be used.

```
| tstats summariesonly=false allow_old_summaries=true count, min(_time) AS firstTime, max(_time) AS lastTime FROM datamodel=Endpoint.Processes WHERE (*Processes.process*="whoami") *Processes.
```
To complete this process, your deployment needs to ingest information on process that include the name of the process responsible for the changes. You should also ensure you are ingesting normalized endpoint data, populating the Endpoint data model in the Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.

This search identifies a non-standard parent process (not matching CMD, PowerShell, or Explorer) spawning ipconfig.exe or systeminfo.exe. This behavior has been seen in FIN7's JSSLoader .NET payload. This is also typically seen when an adversary is injected into another process performing different discovery techniques.

This event is a threat signature since these tools are commonly executed with a shell application or Explorer parent, and not by another application.

This search might return false positives since automated tools used by a system administrator can also exhibit this behavior.

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. If you are using Sysmon, you must have at least version 6.0.4 of the Sysmon TA. You should also ensure you are ingesting normalized endpoint data, populating the Endpoint data model in the Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.

This search detects executions of JScript using a CScript process. Commonly, when a user runs a JScript file, it is executed by the wscript.exe application.
This technique is seen in FIN7 attacks, which execute malicious script using the CScript process. This behavior is uncommon in benign scenarios.

```splunk
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

► **MS scripting process loading LDAP module**

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. If you are using **Sysmon**, you must have at least version 6.0.4 of the Sysmon TA.

This search detects a suspicious MS scripting process, such as wscript.exe or cscript.exe, that loads a LDAP module to process a LDAP query.

This technique is seen in FIN7 attacks where it uses Javascript to execute a LDAP query to parse host information that will send to its C2 server.

False positives from this search might occur since automation scripting language can used by a network operator to perform a LDAP query. Tune and filter known instances where a renamed rundll32.exe might be used.

```splunk
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

► **MS scripting process loading WMI module**

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. If you are using **Sysmon**, you must have at least version 6.0.4 of the Sysmon TA.

This search detects a suspicious MS scripting process, such as wscript.exe or cscript.exe, that loads a WMI module to process a WMI query.

This technique is seen in FIN7 attacks where it uses Javascript to execute a WMI query to parse host information.

The information provided in Splunk Lantern is intended for informational and educational purposes only. All information is provided in good faith, however, Splunk disclaims any and all representations and warranties, express and implied, regarding the information provided, including without limitation any warranties and representations regarding the completeness, adequacy or accuracy of the information. You agree to take full responsibility for the results arising from the use of the information provided.
that will send to its C2 server.

False positives from this search might occur since automation scripting language can used by a network operator to perform a WMI query. Tune and filter known instances where a renamed rundll32.exe might be used.

```
```

► Non-Chrome process accessing Chrome default directory

To complete this process, your deployment needs to ingest Windows Security Event logs and should also track event code 4663. To enable code 4663, enable "Audit Object Access" in your Group Policy, then check the two boxes listed for both "Success" and "Failure."

This search detects a non-Chrome process accessing the files in the Chrome user default folder. This folder contains the SQLite database of the Chrome browser related to a user's login, history, cookies, and other user data. This process is performed by the FIN7 JSSLoader to collect information on the compromised host.

```
| search (EventCode=4663 Object_Name="*\Google\Chrome\User Data\Default*" eventtype=wineventlog_security NOT process_name="*\chrome.exe" NOT process_name="*\explorer.exe" NOT process_name="*sql*") | stats count min(_time) AS firstTime max(_time) AS lastTime BY Object_Name Object_Type process_name Access_Mask Accesses process_id EventCode dest user | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime) | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

► Non-Firefox process accessing Firefox profile directory

To complete this process, your deployment needs to ingest Windows Security Event logs and should also track event code 4663. To enable code 4663, enable "Audit Object Access" in your Group Policy, then check the two boxes listed for both "Success" and "Failure."

This search detects a non-Firefox process accessing the files in the Firefox profile folder. This folder contains the SQLite database of the Firefox browser related to a user's login, history, cookies and other user data. This process is performed by the FIN7 JSSLoader to collect information on the compromised host.

```
| search (EventCode=4663 Object_Name="*\AppData\Roaming\Mozilla\Firefox\Profiles*" eventtype=wineventlog_security NOT process_name="*\explorer.exe" NOT process_name="*\firefox.exe" NOT process_name="*sql*")
```

The information provided in Splunk Lantern is intended for informational and educational purposes only. All information is provided in good faith, however, Splunk disclaims any and all representations and warranties, express and implied, regarding the information provided, including without limitation any warranties and representations regarding the completeness, adequacy or accuracy of the information. You agree to take full responsibility for the results arising from the use of the information provided.
MS Office application creating executable

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. If you are using Sysmon, you must have at least version 6.0.4 of the Sysmon TA. You should also ensure you are ingesting normalized endpoint data, populating the Endpoint data model in the Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.

This search detects a suspicious MS Office application that creates executables or scripts in the host. This behavior is commonly seen in spear phishing attacks using MS Office attachments, where malicious files or scripts are created to compromise the host.

False positives from this search might occur since a normal macro can drop scripts or tools as part of automation. However, this behavior is still suspicious as it is not commonly seen, so should it be investigated further.

Office product spawning wmic.exe

To complete this process, your deployment needs to ingest information on process that include the name of the process responsible for the changes from your endpoints. You should also ensure you are ingesting normalized endpoint data, populating the Endpoint data model in the Processes node in the Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.
This search identifies Office suite applications used as a parent process spawning wmic.exe.

If you return potential positive results from this search, investigate further by reviewing all file modifications, as well as capture and analyze any artifacts on disk. If you find that the Office product or wmic.exe has reached out to a remote destination, capture and block the IPs or domain, and review additional parallel processes for further activity.

```
| rename "Processes.*" AS "***"
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

#### VBScript execution using WScript app

To complete this process, your deployment needs to ingest information on process that include the name of the process responsible for the changes from your endpoints. You should also ensure you are ingesting normalized endpoint data, populating the [Endpoint data model](#) in the Processes node in the Common Information Model (CIM).

For information on installing and using the CIM, see the [Common Information Model documentation](#).

This search detects a suspicious WScript command used to execute VBScript. This is a technique used by malware to execute malicious VBS files using the WScript application, intended to evade process parent child detections or AV script emulation systems.

```
| rename "Processes.*" AS "***"
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

#### WScript or CScript suspicious child process

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. You should also ensure you are ingesting normalized endpoint data, populating the [Endpoint data model](#) in the Processes node in the Common Information Model (CIM). For information on installing and using the CIM, see the [Common Information Model documentation](#).

This search identifies a suspicious spawned process by WScript or CScript. This technique is a commonly used by malware to execute different LOLBin, other scripts like PowerShell, or to spawn a suspended process to inject its code as a defense evasion.
False positives from this search might occur since the search can detect some normal script that uses several application tools that are in the list of the child process it detects. Administrators might also create VBScript or JS script that uses several tools as part of its execution. Filter as needed.


► XSL script execution with WMIC

To complete this process, your deployment needs to ingest logs with the process name, parent process, and command-line executions from your endpoints. You should also ensure you are ingesting normalized endpoint data, populating the Endpoint data model in the Processes node in the Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.

This search detects a suspicious wmic.exe process or renamed WMIC process to execute a malicious XSL file. This technique is used by FIN7 malware to execute its malicious JScript using an .xsl file as the loader with the help of the wmic.exe process.


Next steps

The content in this article comes from Splunk Enterprise Security (ES). As a Splunk premium security solution, ES solves a wide range of security analytics and operations use cases including continuous security monitoring, advanced threat detection, compliance, incident investigation, forensics and incident response. Splunk ES delivers an end-to-end view of an organization’s security posture with flexible investigations, unmatched performance, and the most flexible deployment options offered in the cloud, on-premises, or hybrid deployment models. If you have questions about this use case, see the Security Research team’s support options on GitHub.

In addition, these Splunk resources might help you understand and implement this use case:

- Use case: Detecting REvil ransomware infections
• Use case: Detecting DarkSide ransomware
• Use case: Detecting ransomware attacks
• Use case: Investigating ransomware attacks
• Blog: Detecting Remcos tool used by FIN7 with Splunk