Detecting IcedID attacks

IcedID is a banking trojan campaign. This malware is seen in Windows environments where it initially uses MS Office as a weapon or attack vector to infect machines, and then targets browsers such as Firefox and Chrome to steal online banking information.

It is known for its unique payload downloaded in C2, where its .png file hides the core shellcode bot using stenography techniques or gzip dat files that contain "license.dat" - the actual core IcedID bot.

Required data

- Endpoint data

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

▶ Account discovery commands

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 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 potential account discovery series of commands used by malware to recon a target machine. Malware using this technique commonly runs a specific series of command processes, or drops a module that runs the series of commands. Searches that show these commands being used are a good indicator of attack if seen in machines used by a non-technical user or department.

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 execution of the chcp.exe application. This utility is used to change the active code page of the console, and is a technique used by IcedID attackers to find out the local region, language, or country of the compromised host.

False positives from this search might occur since other tools or scripts can use this application to change a code page to UTF-8* or others. Tune and filter as necessary.

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 Processes node of 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 command-line arguments where cmd.exe /c is used to run a program. cmd /c is used to run commands in MS-DOS and terminates after the command or process completion. This technique is used by attackers to run batch commands using a different shell like PowerShell or processes other than cmd.exe.

False positives from this search might be high, based on legitimate scripted code in any environment. Tune and filter as necessary.
Create remote thread in shell application

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 process injection in the command shell. This technique is seen in IcedID attacks where it runs a cmd.exe process to inject its shellcode as part of its execution.

```
| search (EventCode=8 (TargetImage="*\cmd.exe" OR TargetImage="*\powershell") (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational))
| stats count min(_time) AS firstTime max(_time) AS lastTime BY TargetImage TargetProcessId SourceProcessId EventCode StartAddress SourceImage Computer
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

Disable scheduled tasks

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 command line used to disable existing scheduled tasks. This technique is used by malware like IcedID to disable security applications in targeted hosts in order to evade detections.

False positives from this search might occur since admins might disable problematic scheduled tasks. Tune and filter as necessary.

```
```
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 the dropping of a suspicious file named license.dat in %appdata%. This behavior is seen in IcedID malware, which contains the core bot that's injected in other processes to steal banking information.

```
| search (EventCode=11 TargetFilename="\license.dat" (TargetFilename="\appdata\*" OR TargetFilename="\programdata\*") (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational))
| stats count min(_time) AS firstTime max(_time) AS lastTime BY TargetFilename EventCode process_id process_name Computer
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

### Exfiltrated archived file creation

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 the creation of the suspicious files, passff.tar and cookie.tar. These files are possible archives of stolen browser information, like history and cookies, in a compromised machine with IcedID.

```
| search (EventCode=11 (TargetFilename="\cookie.tar" OR TargetFilename="\passff.tar") (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational))
| stats count min(_time) AS firstTime max(_time) AS lastTime BY TargetFilename EventCode process_id process_name Computer
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

### Mshta.exe spawning rundll32 or regsvr32 process

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 Common Information Model (CIM). For information on installing and using the CIM, see the Common Information Model documentation.
This search detects a suspicious mshta.exe process that spawns rundll32 or regsvr32 child processes. This technique is seen in malware attacks to initiate .dll stage loaders that run and download the malware payload.

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

---

**NLTest.exe domain trust discovery**

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

This search looks for the execution of NLTest.exe with command-line arguments utilized to query for domain trust information. Two arguments are used in this search:

- **/domain trusts** returns a list of trusted domains
- **/all_trusts** returns all trusted domains.

Attackers use NLTest.exe to enumerate the current domain to help them understand where to pivot next.

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

---

**MS Office applications spawning regsvr32 or rundll32 processes**

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 suspicious spawned processes of MS Office applications due to macro or malicious code. This technique is seen in IcedID malware which uses MS Office as a weapon or attack vector to infect machines.

```
| tstats summariesonly=false allow_old_summaries=true count, min(_time) AS firstTime, max(_time) AS lastTime FROM datamodel=Endpoint.Processes WHERE ("Processes.process_name"=" mshta.exe"
```

---

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

Make the following adjustments to the search above to search for Regsvr32 or rundll32 processes.

<table>
<thead>
<tr>
<th>Process to search for</th>
<th>Replacement search lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regsvr32</td>
<td>(&quot;Processes.process_name&quot;=regsvr32.exe OR &quot;Processes.original_file_name&quot;=REGSVR32.EXE))</td>
</tr>
<tr>
<td>Rundll32</td>
<td>(&quot;Processes.process_name&quot;=rundll32.exe OR &quot;Processes.original_file_name&quot;=RUNDLL32.EXE))</td>
</tr>
</tbody>
</table>

► MS Office document running macro code

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

This search identifies suspicious MS Office documents that use macro code. Macro code is a prevalent malware attack vector that runs malicious payloads, downloads malware payloads, or other malware components.

It is a good practice to disable macros by default to avoid automatically running macro code while opening or closing a MS Office document.

False positives from this search might occur since macros can be used in normal MS Office documents for automation purposes.

| search (EventCode=7 (ImageLoaded="\\\VBE7.DLL" OR ImageLoaded="\\\VBE7INTL.DLL" OR ImageLoaded="\\\VBEUI.DLL") (process_name="EXCEL.EXE" OR process_name="POWERPNT.EXE" OR process_name="WINWORD.EXE") (source=Syslog:Linux-Sysmon/Operational OR source=XmWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourceType=XmWinEventLog:Microsoft-Windows-Sysmon/Operational)) | stats min(_time) AS firstTime max(_time) AS lastTime values(ImageLoaded) AS AllImageLoaded count BY Computer EventCode Image process_name ProcessId ProcessGuid |
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime) | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime) |

► MS Office product spawning mshta.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 any MS Office product spawning `mshta.exe`. In malicious instances, the command-line of `mshta.exe` contains a HTA file locally, or a URL to the remote destination.

If you see positive results from this search, review all file modifications, and capture and analyze any artifacts on disk. The Office product or `mshta.exe` will have reached out to a remote destination and you will need to capture and block the IPs or domain. You should also review additional parallel processes for further activity.

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

**Registry keys used for persistence**

To complete this process, your deployment needs to ingest normalized data that records registry activity from your hosts to populate the **Endpoint data model** in the Registry node in the Common Information Model (CIM). This is typically populated via endpoint detection-and-response products, such as **Carbon Black** or endpoint data sources, such as **Sysmon**. The data used for this search is typically generated via logs that report reads and writes to the registry. For information on installing and using the CIM, see the **Common Information Model documentation**.

This search looks for modifications to registry keys that can be used to launch an application or service at system startup.

```
| tstats summariesonly=false allow_old_summaries=true count FROM datamodel=Endpoint.Registry WHERE ("Registry.registry_path"="\currentversion\run" OR "Registry.registry_path"="\currentversion\Windows\Appinit_Dlls" OR "Registry.registry_path"="\CurrentVersion\Winlogon\Shell" OR "Registry.registry_path"="\CurrentVersion\Winlogon\Notify" OR "Registry.registry_path"="\CurrentVersion\Winlogon\Userinit" OR "Registry.registry_path"="\CurrentVersion\Winlogon\VmApplet" OR "Registry.registry_path"="\currentversion\policies\explorer\run" OR "Registry.registry_path"="HKLM\SOFTWARE\Microsoft\Netsh\*" OR "Registry.registry_path"="HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\*" "Registry.registry_key_name"=Debugger) OR ("Registry.registry_path"="\CurrentControlSet\Control\Lsa" "Registry.registry_key_name"=Security Packages) OR ("Registry.registry_path"="\CurrentControlSet\Control\Lsa\OSConfig" "Registry.registry_key_name"=Security Packages) OR ("Registry.registry_path"="\\Microsoft\Windows NT\CurrentVersion\SilentProcessExit\*" OR ("Registry.registry_path"="\\currentVersion\Windows" "Registry.registry_key_name"=Load)) OR ("Registry.registry_path"="\\CurrentVersion" "Registry.registry_key_name"="Svchost") OR ("Registry.registry_path"="\\CurrentControlSet\Control\Session Manager" "Registry.registry_key_name"="BootExecute") OR ("Registry.registry_path"="\\Software\Run" "Registry.registry_key_name"="auto_update") BY _time span=1h, "Registry.dest", "Registry.user", "Registry.registry_path",
```

---

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.
Regsvr32 with known silent switch command line

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 regsvr32.exe utilizing a silent switch to load DLLs. This technique is seen in IcedID campaigns to load an initial DLL that will download the second stage loader that will then download and decrypt the config payload.

The switch type might be either a hyphen - or forward slash /. This behavior is typically found with -s, and it is possible there are more switch types used.

Rundll32 create remote thread to a process

To complete this process, your deployment needs to ingest logs with the SourceImage, TargetImage, and EventCode executions from your endpoints related to create remote thread or injecting codes. If you are using Sysmon, you must have at least version 6.0.4 of the Sysmon TA.

This search identifies a suspicious remote thread execution of rundll32.exe process to a cmd.exe process. This technique is seen in IcedID malware to run malicious code in a normal process, for the purpose of defense evasion.
and to steal sensitive information from the compromised host.

| search (EventCode=8 SourceImage="\rundll32.exe" TargetImage="*.exe" (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational)) | stats count min(_time) AS firstTime max(_time) AS lastTime BY SourceImage TargetImage TargetProcessId SourceProcessId StartAddress EventCode Computer | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime) | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime) |

► Rundll32 create remote thread in browser

To complete this process, your deployment needs to ingest logs with the SourceImage, TargetImage, and EventCode executions from your endpoints related to creating remote thread or injecting codes. If you are using Sysmon, you must have at least version 6.0.4 of the Sysmon TA.

This search identifies a suspicious remote thread execution of the rundll32.exe process in the Firefox, Chrome, or Internet Explorer browsers. This technique is seen in IcedID malware where it hooks the browser to parse banking information.

| search (EventCode=8 SourceImage="\rundll32.exe" (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational) (TargetImage="\chrome.exe" OR TargetImage="\firefox.exe" OR TargetImage="\iexplore.exe" OR TargetImage="\microsoftedgecp.exe")) | stats count min(_time) AS firstTime max(_time) AS lastTime BY SourceImage TargetImage TargetProcessId SourceProcessId StartAddress EventCode Computer | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime) | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime) |

► Rundll32 process creating .exe DLL files

To complete this process, your deployment needs to ingest logs with the process name, TargetFilename, and event code 11 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 rundll32 process that drops executable (.exe or .dll) files. This behavior is seen in IcedID where it tries to drop a copy of itself in a temp folder, or tries to download an executable and drop it in either the appdata or programdata folders as part of its execution.

| search (EventCode=11 process_name="rundll32.exe" (TargetFilename="*.dll" OR TargetFilename="*.exe") (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational)) | stats count min(_time) AS firstTime max(_time) AS lastTime BY Image TargetFilename ProcessGuid dest user_id | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime) | convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime) |

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.
SQLite module in temp folder

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 file creation of sqlite3.dll in the %temp% folder. This behavior is seen in IcedID malware where it downloads a SQLite module to parse the Chrome or Firefox browser database to steal browser information related to bank, credit card, or user credentials.

```
| search (EventCode=11 TargetFilename="*\temp\*" (TargetFilename="*\sqlite32.dll" OR TargetFilename="*\sqlite64.dll") (source=Syslog:Linux-Sysmon/Operational OR source=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational OR sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational))
| stats count min(_time) AS firstTime max(_time) AS lastTime BY process_name TargetFilename EventCode ProcessId Image
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)
```

Suspicious rundll32 command line

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 rundll32.exe command line used to run a DLL file. This technique is seen in IcedID malware used to load its payload DLL with the a parameter to load the license.dat encrypted DLL payload.

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

Suspicious rundll32 plugininit

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 rundll32.exe process with a plugininit parameter. This technique is seen in IcedID malware used to run an initial DLL stager, which then downloads another payload to the compromised...
| rename "Processes.*" AS "*"
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)

---

**Next steps**

The content in this article comes from [Splunk Enterprise Security (ES)](https://www.splunk.com). 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](https://github.com/splunk/security-research).

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

- **E-book:** [Ransomware, malware and cyberthreats](https://www.splunk.com)
- **Blog:** [Hunting with Splunk: The basics](https://www.splunk.com)