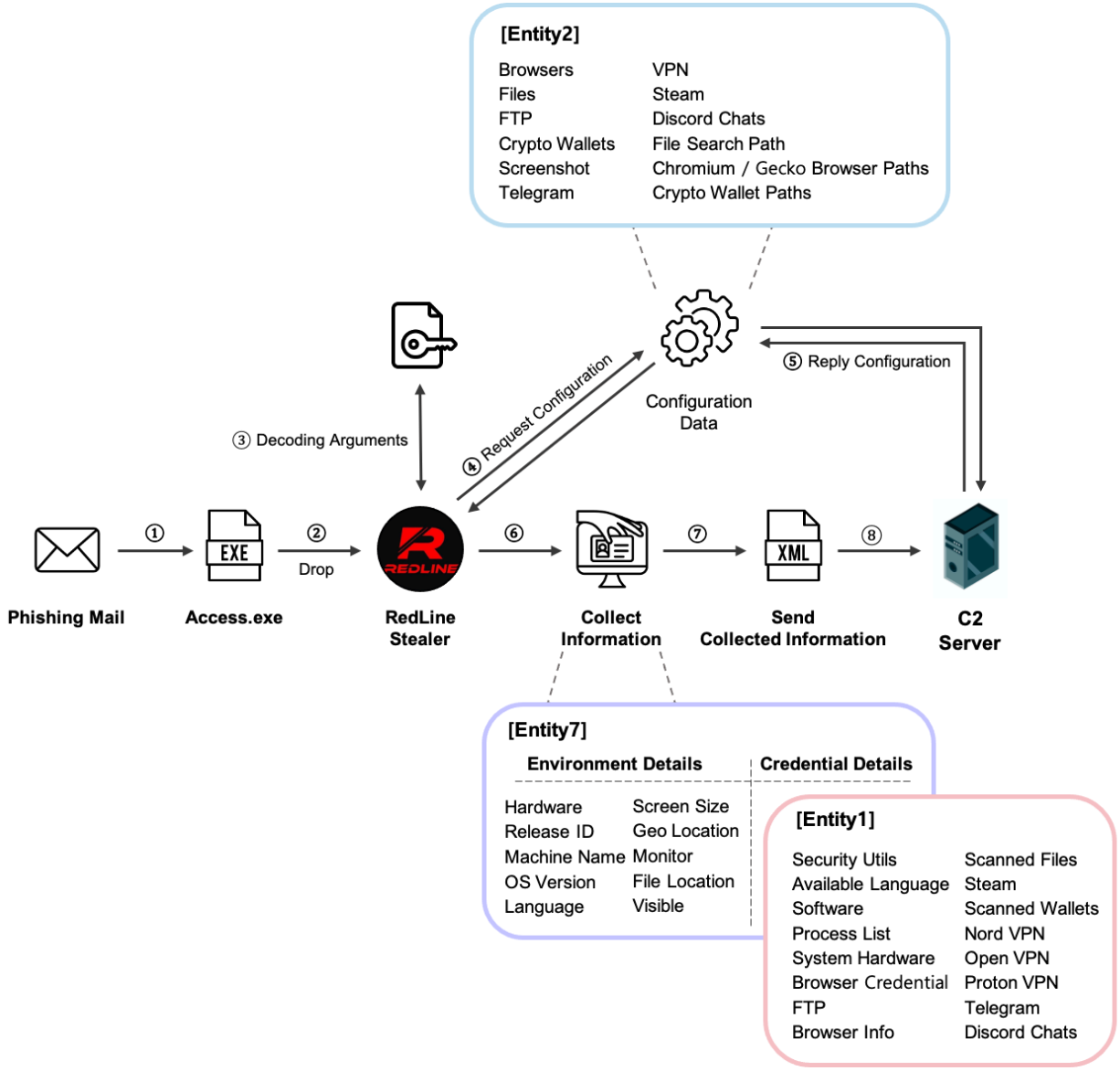
# Threat Hunting: Detecting Browser Credential Stealing [T1555.003]

Archived: 2026-04-02 11:36:50 UTC

Password Managers have seen rapid adoption by organisations as they provide a safe space to store and access your passwords. Native password managers such as Chrome and Edge Password managers offer users a convenient way of creating secure passwords for different sites without the hassle of remembering each password. As the usage of similar passwords across websites goes down, threat actors have adopted and have now begun to target these password managers present in your web browser.

Stealers such as Redline are in the news as they provide a low barrier of entry to new cybercriminals, who then use these credentials to provide initial access to other sophisticated groups.



Redline Stealer Operation: Illustration by Jiho Kim | S2W Talon

## Browser Credential Dumping - MITRE ATT&CK T1555

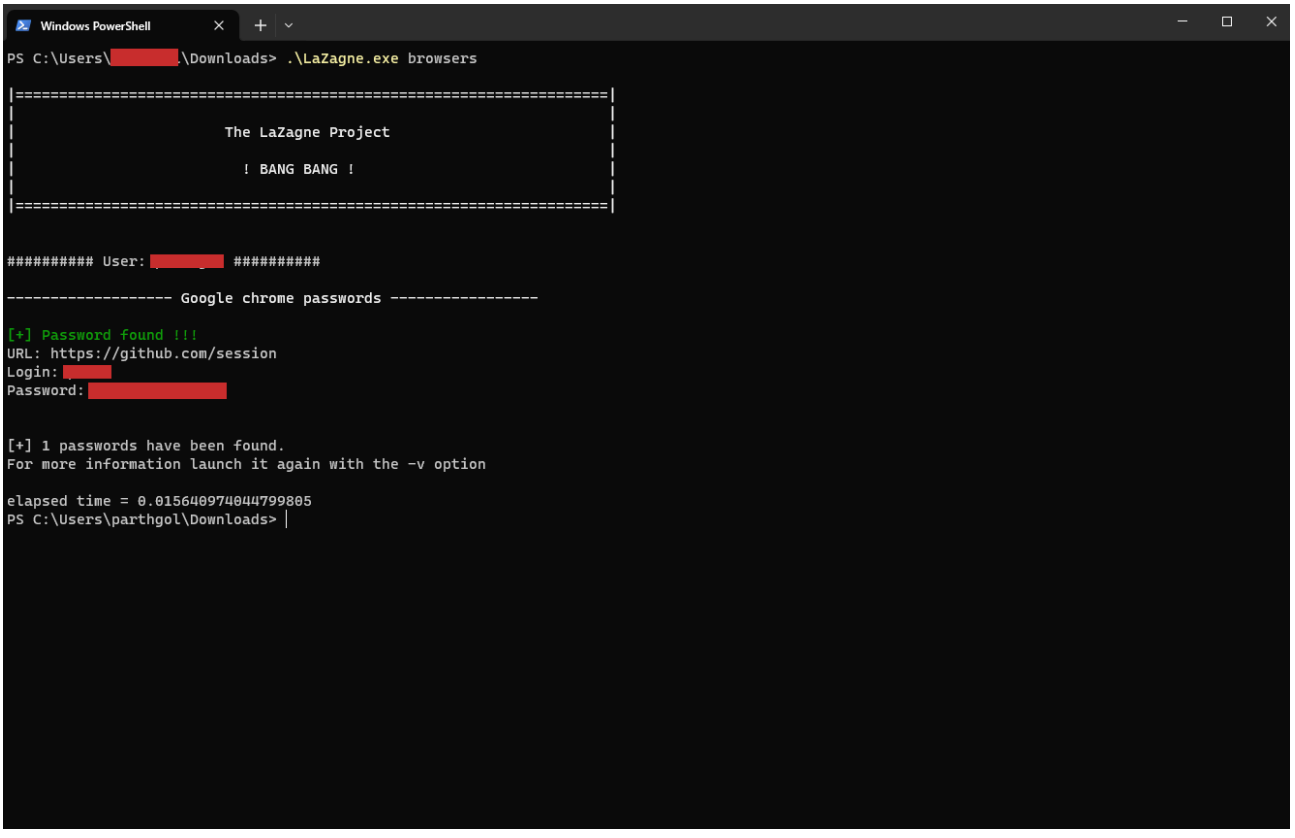
Browser Credential dumping is a technique adversaries use to steal credentials from your browsers. People save login credentials in browsers to make the login process faster. Malware such as Redline Stealer, Zaraza bot, and other info stealers have been actively targeting users and organizations to gain access to browser credentials. These credentials are made available to threat actors who use these credentials to breach various organizations.

This post will showcase how to detect browser credential extraction, weed out false positives, and improve our resilience against this threat.

---

## Tools of the Trade

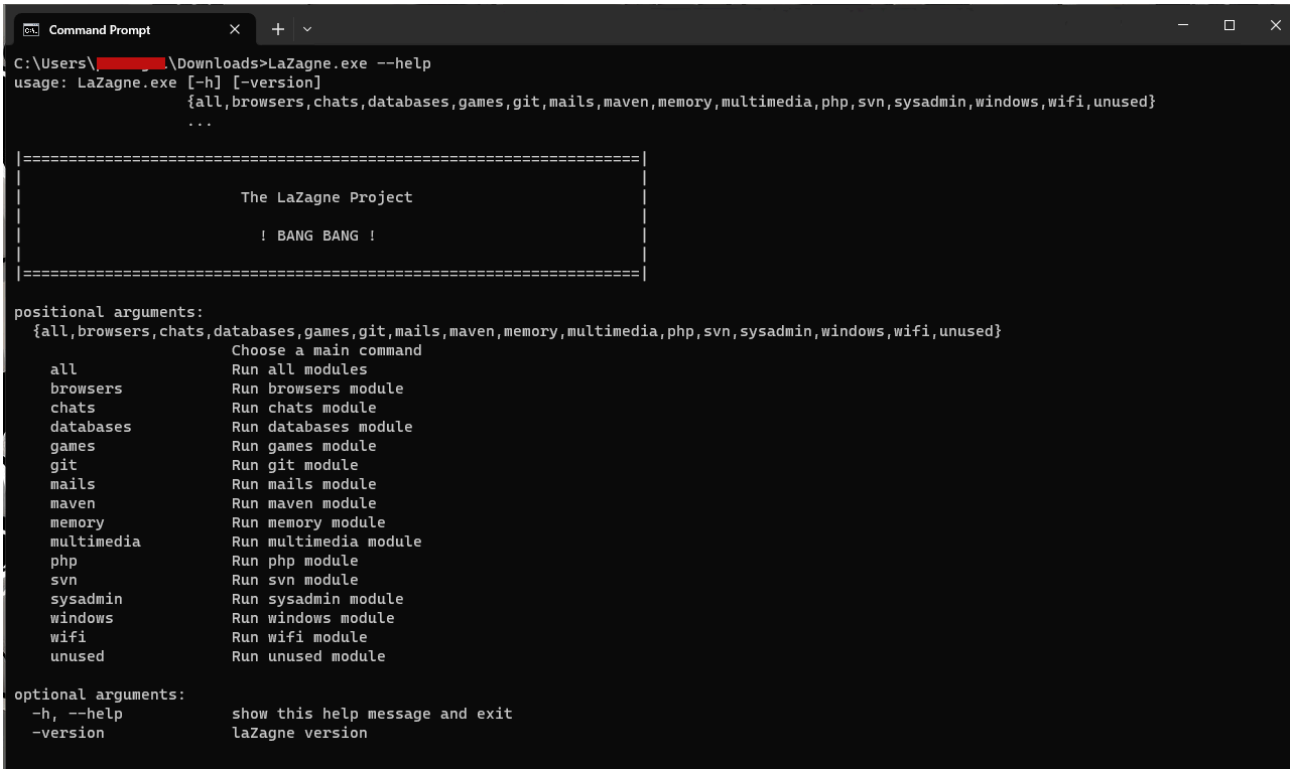
There are various tools, open source and closed, which adversaries use for stealing credentials from browsers. Tools such as Lazagne and HackerBrowserData are open source and provide customizability to advanced attackers, whereas tools such as Nirsoft's WebBrowserPassView are closed source and cannot be modified easily. Direct integration to C2 Frameworks such as Metasploit's post/multi/gather/firefox_creds and post/windows/gather/enum_chrome modules allow quick access to browser passwords for adversaries.




Extraction of Browser passwords using lazagne

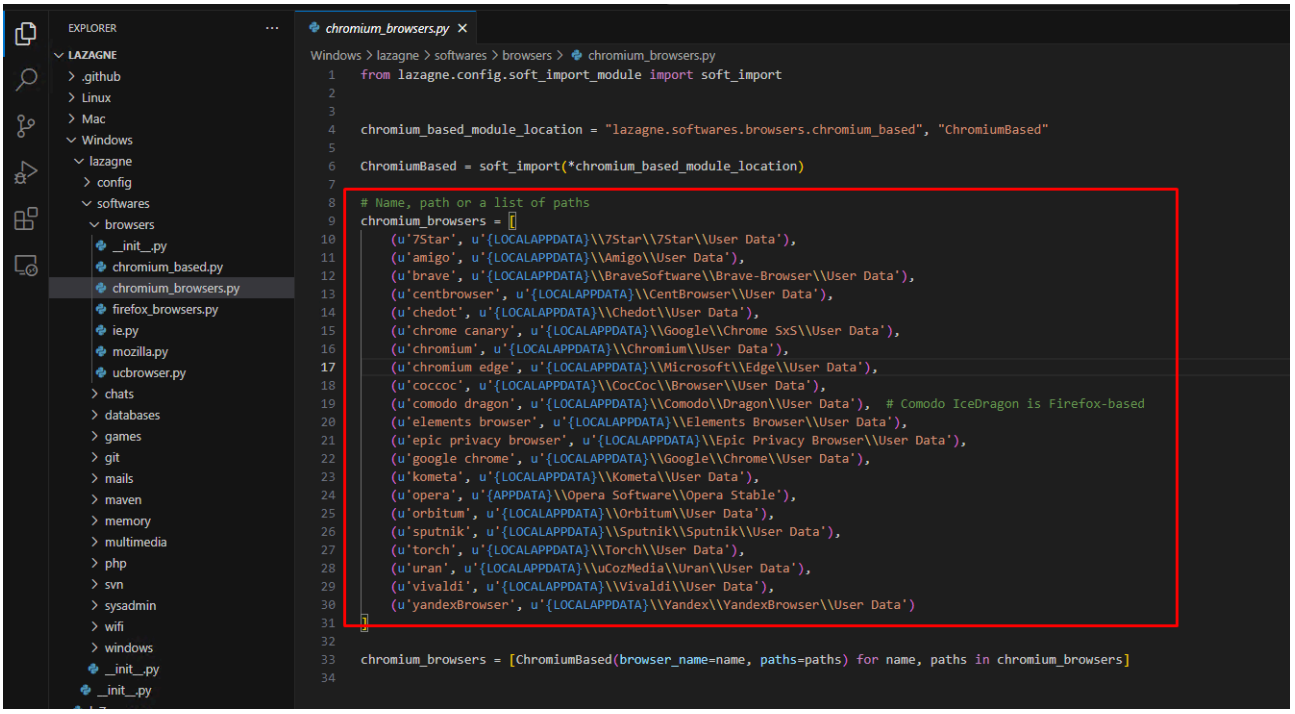
## Methodology

To identify how tools such as Lazagne and HackBrowserData extract browser credentials from a host machine, we can download their source code for examination and find key detection opportunities.

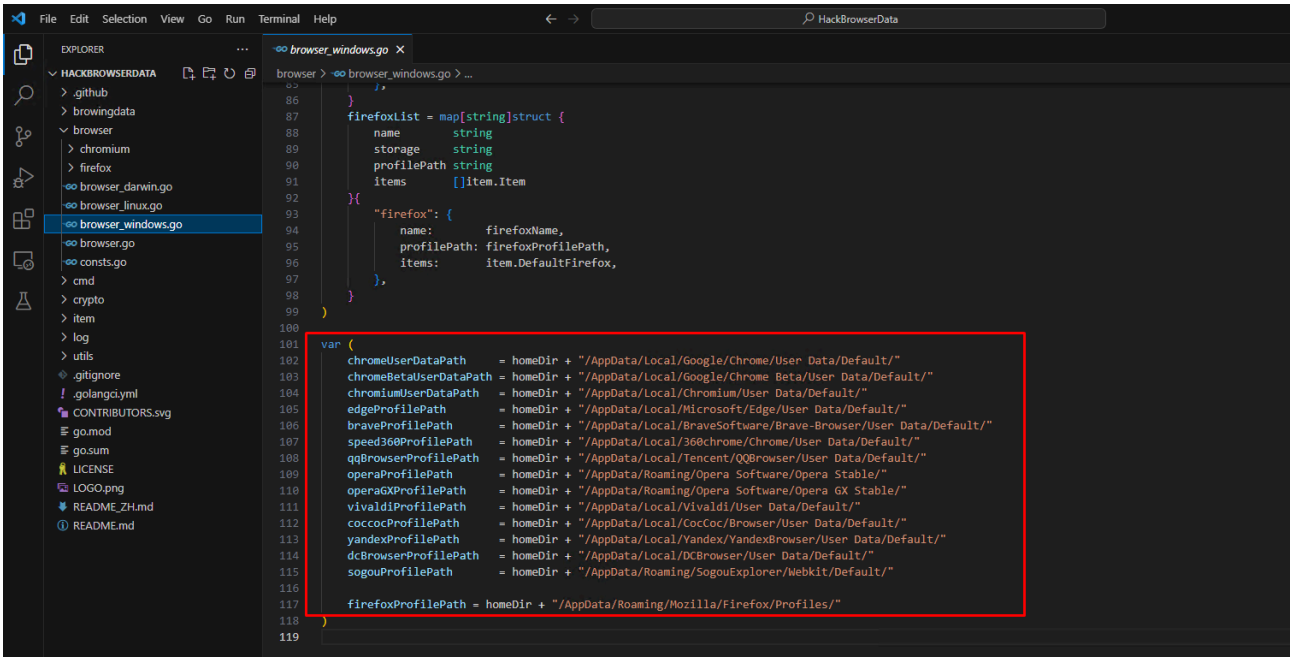



Lazagne help menu

Examining the code for Lazagne and HackBrowserData, it is clear that both tools extract data from predefined file locations in the operating system. Both tools read the following known file paths.

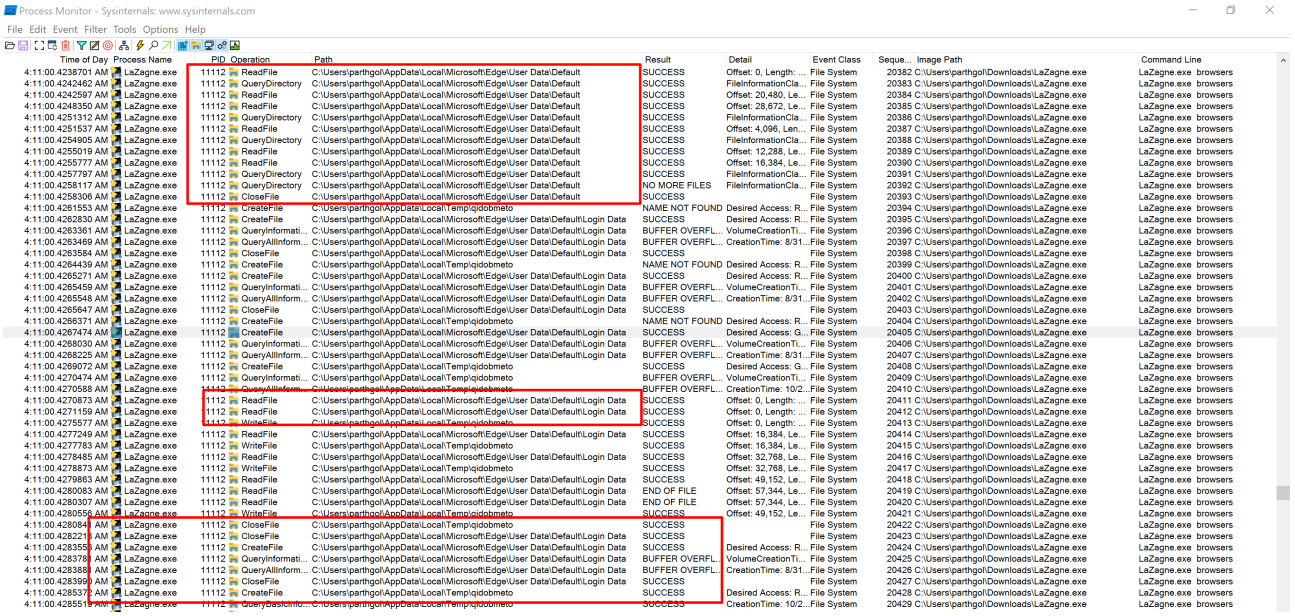



Lazagne source code



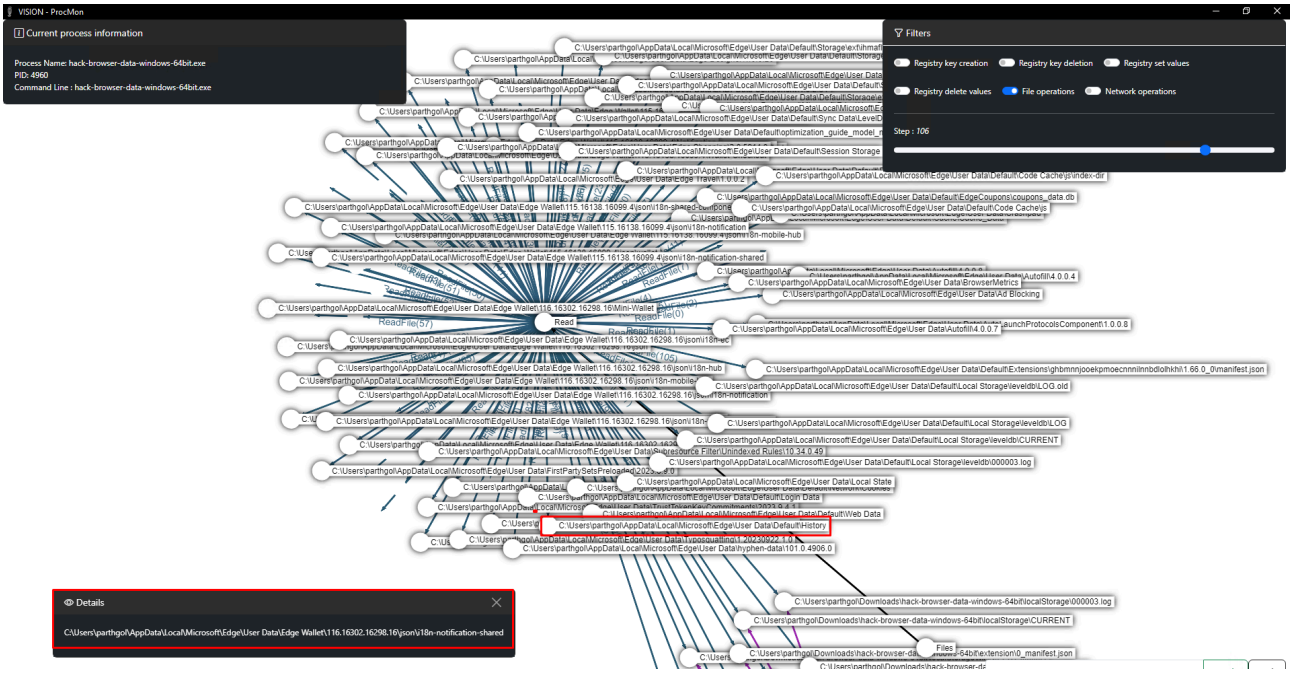

HackBrowserData source code

We can also execute both tools and observe them in [Procmon](#) to further corroborate our findings. Procmon will show us any process creation, registry/file access and other events to help us narrow down key behaviours among browser credential extraction tools.




Lazagne process view in Procmon

We can also visualise the Procmon logs using [Vision-ProcMon](#), which allows for a graphical view of operations such as file access and modification of registry keys. Utilising the Step Option in Vision-Procmon, we can trace the events and identify multiple paths used to dump browser credentials.




hackbrowserdata procmon log graph view in Vision-procmon

Analysing the procmon logs of various open and closed source tools, we can confirm that all tools access fixed paths where the browsers store their data(such as cookies, credentials, history) and then process these files to extract credentials.

---

## Detecting Unauthorized Access to Browser Files

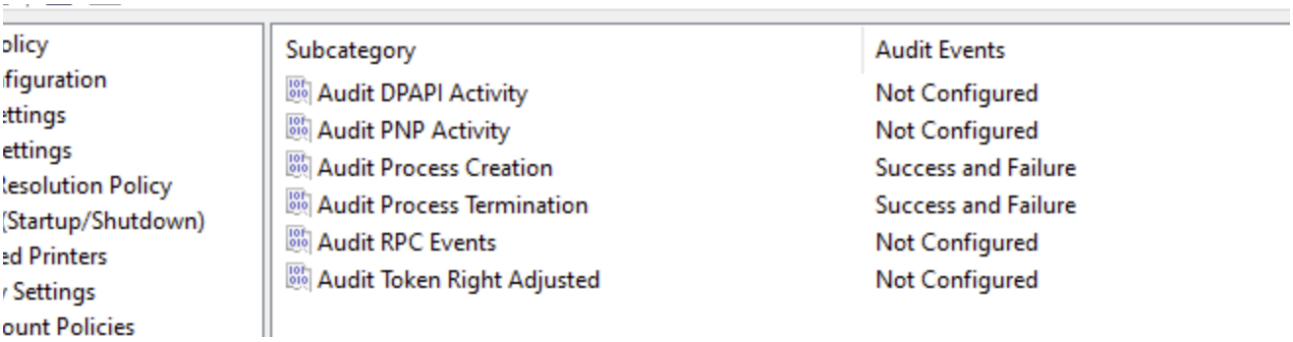
To set up correct monitoring and detection of browser credential extraction, we need to enable auditing features in Windows to receive logs. We need to get process creation logs to monitor for known malicious command lines and file access logs to monitor unauthorised access to browser files.

## Enabling Process Creation Event Logs

Enabling Process Creation auditing will create Event ID 4688 and other necessary details such as Process Path, Parent, Command line, etc, using which we can monitor for malicious command lines. We will use Group Policy Editor to set up Process Creation Auditing.

### Configuring Process Auditing:

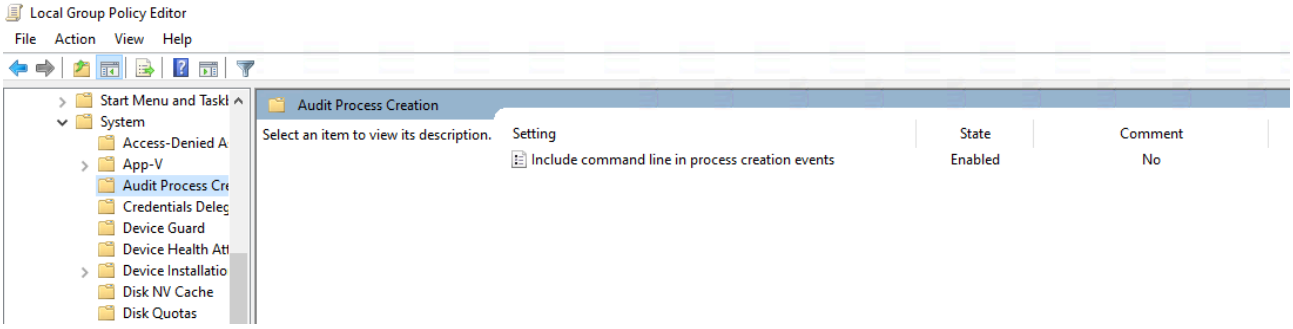
1. Computer Configuration > Windows Settings > Security Settings > Advanced Audit Policy Configuration > Detailed Tracking
2. Select: Audit Process Creation, Select: Success + Failure, Select: OK




Local Group Policy Editor

### Configuring Command Line in Process Auditing:

1. Computer Configuration > Policies > Administrative Templates > System > Audit Process Creation
2. Select: Include command line in process creation events, Select Enabled, and Press OK

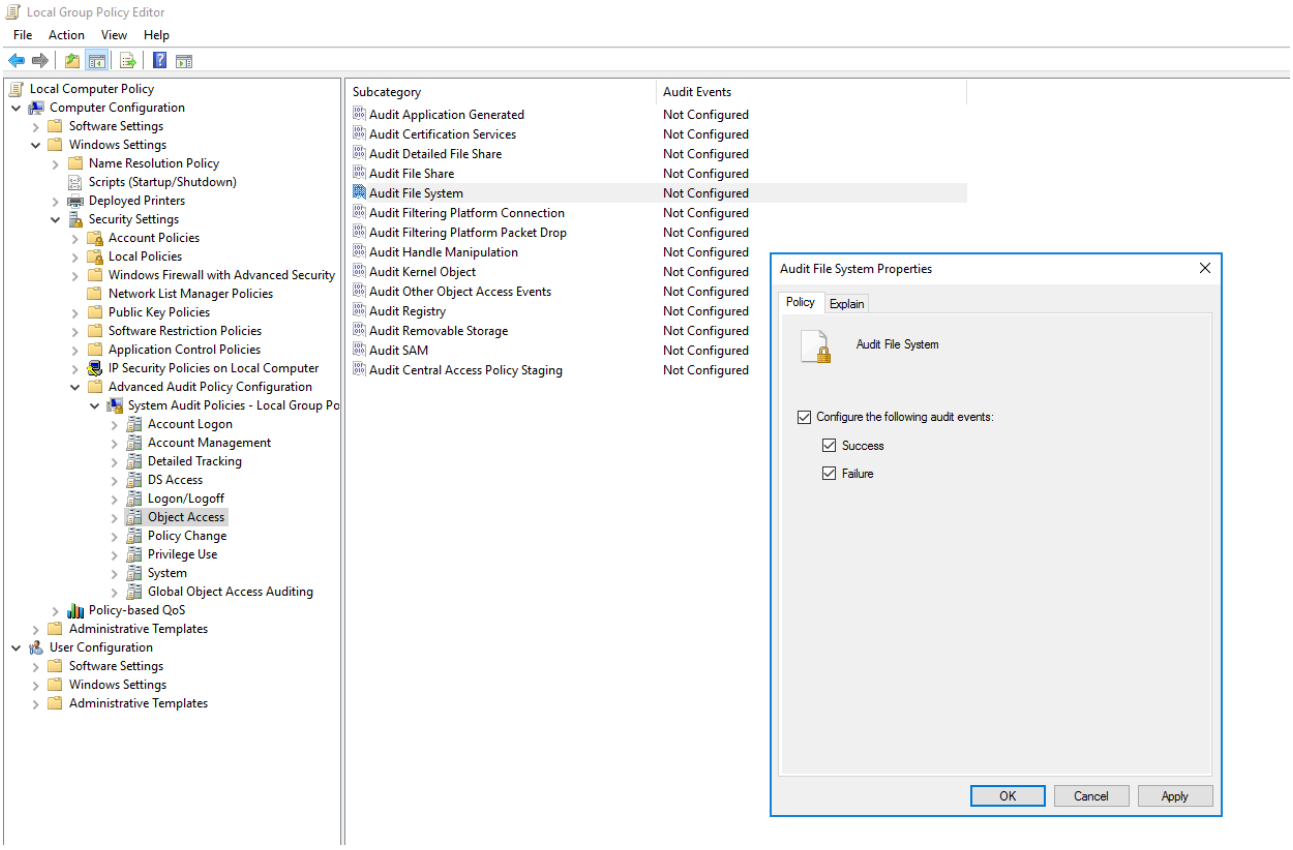



Local Group Policy Editor

## Enabling File Access Audit Logs

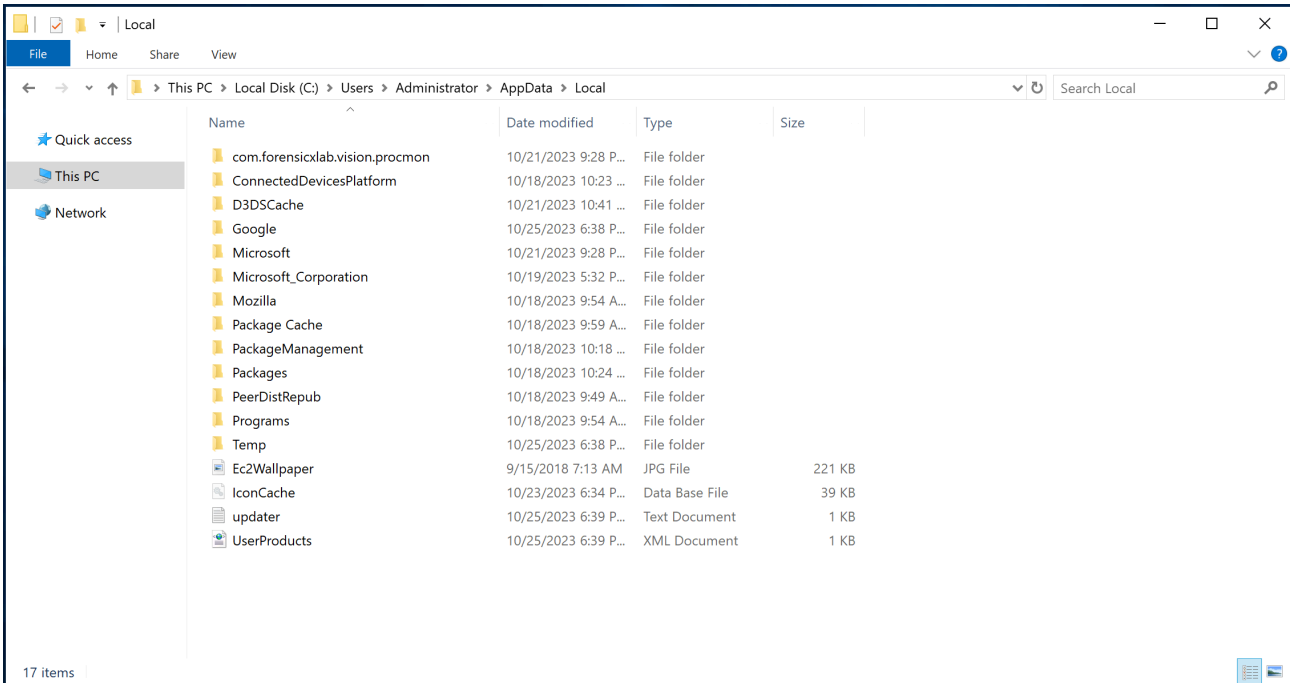
Enabling File System auditing is a two-step process where we first enable auditing in group policy and then configure individual files/folders we want to monitor. Configuring File Access auditing will create Event ID 4663 along with other necessary details such as Object Path and which process is accessing the files. We will use Group Policy Editor to set up Process Creation Auditing.

Computer Configuration -> Policies -> Windows Settings -> Security Settings -> Advanced Audit Policy Configuration -> Audit Policies -> Object Access -> Audit File System



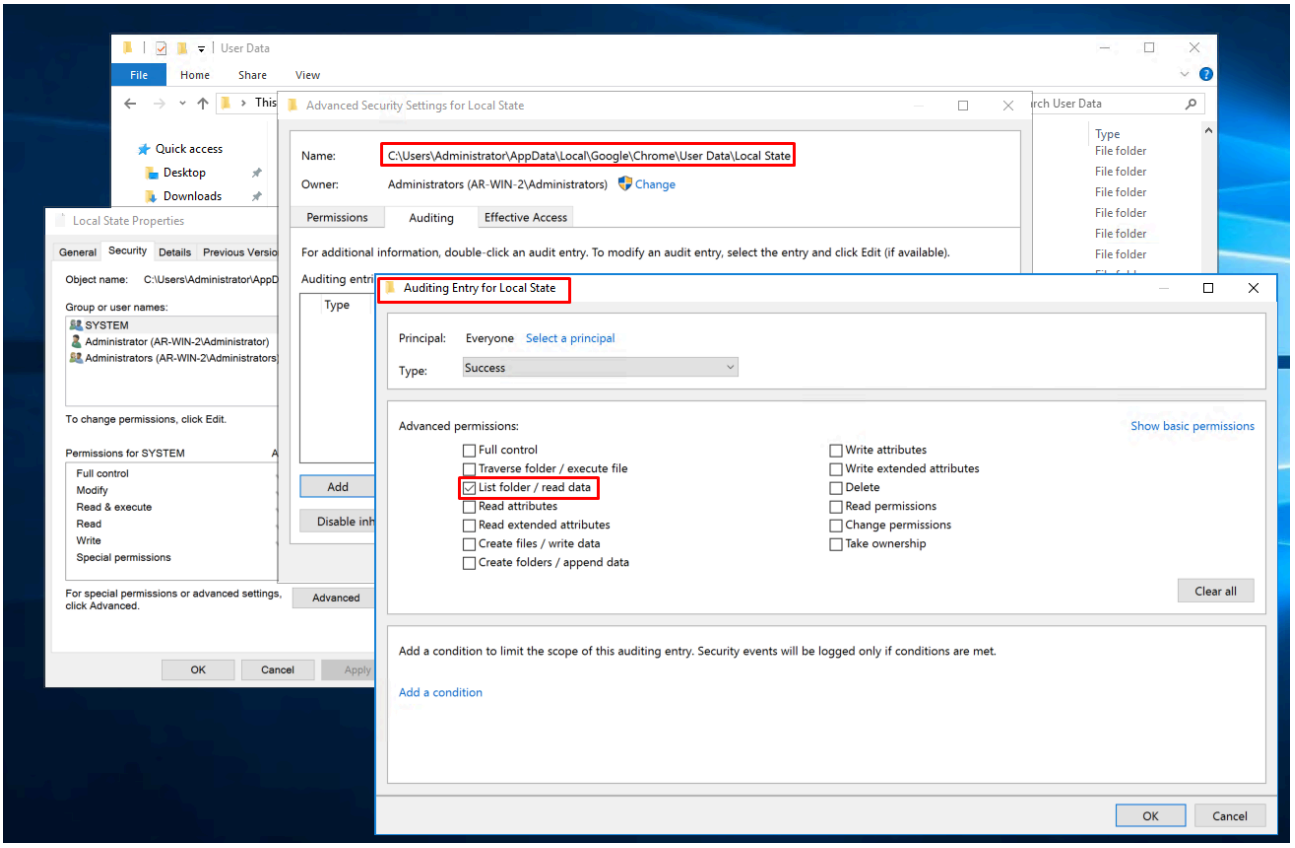

Local Group Policy Editor

1. Navigate to your Local Appdata folder at %LOCALAPPDATA% and configure auditing for each browser folder.




Browser impotant Files / Folder directory

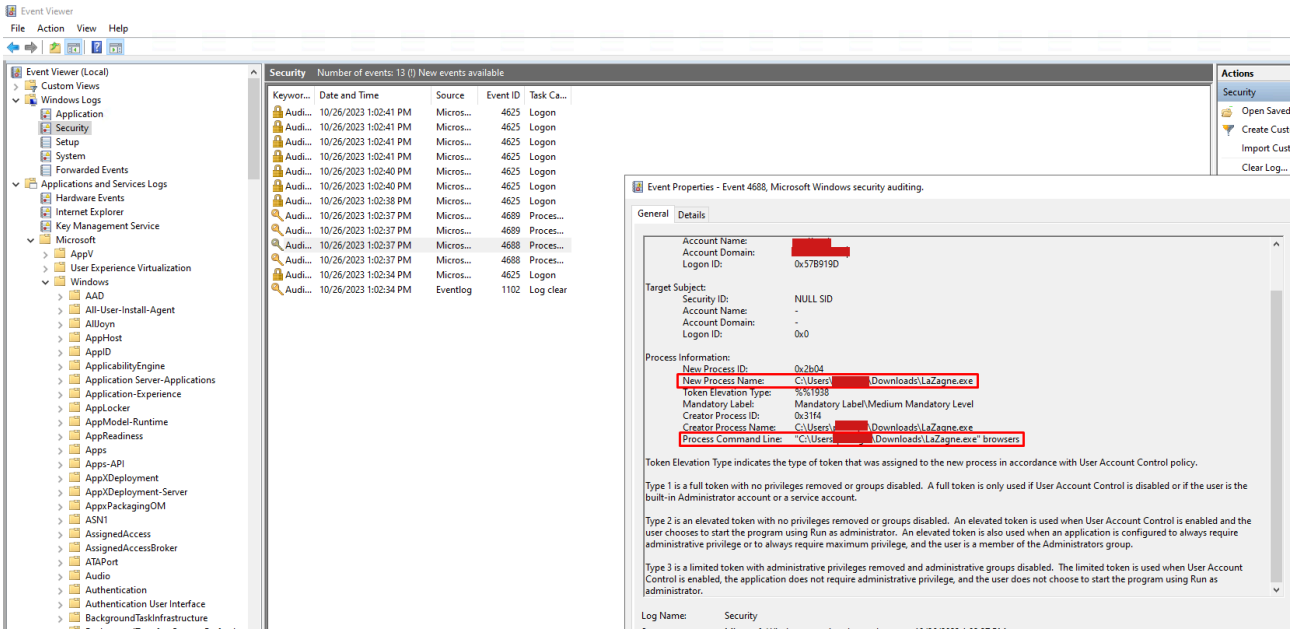
1. Right-click the target file/folder and select "**Properties**"
2. Select "**Security**" tab and click "**Advanced**"
3. Select "**Auditing**" tab and click "**Continue**"
4. Click "**Add**" to insert a new auditing entry
5. Click "**Select a principal**" and insert "**Everyone**"
6. Click "**Clear all**" in the permissions and click "**Show advanced permissions**
7. Tick "**List folder / read data**"
8. Save all the changes




Enable Auditing for List folder / read data

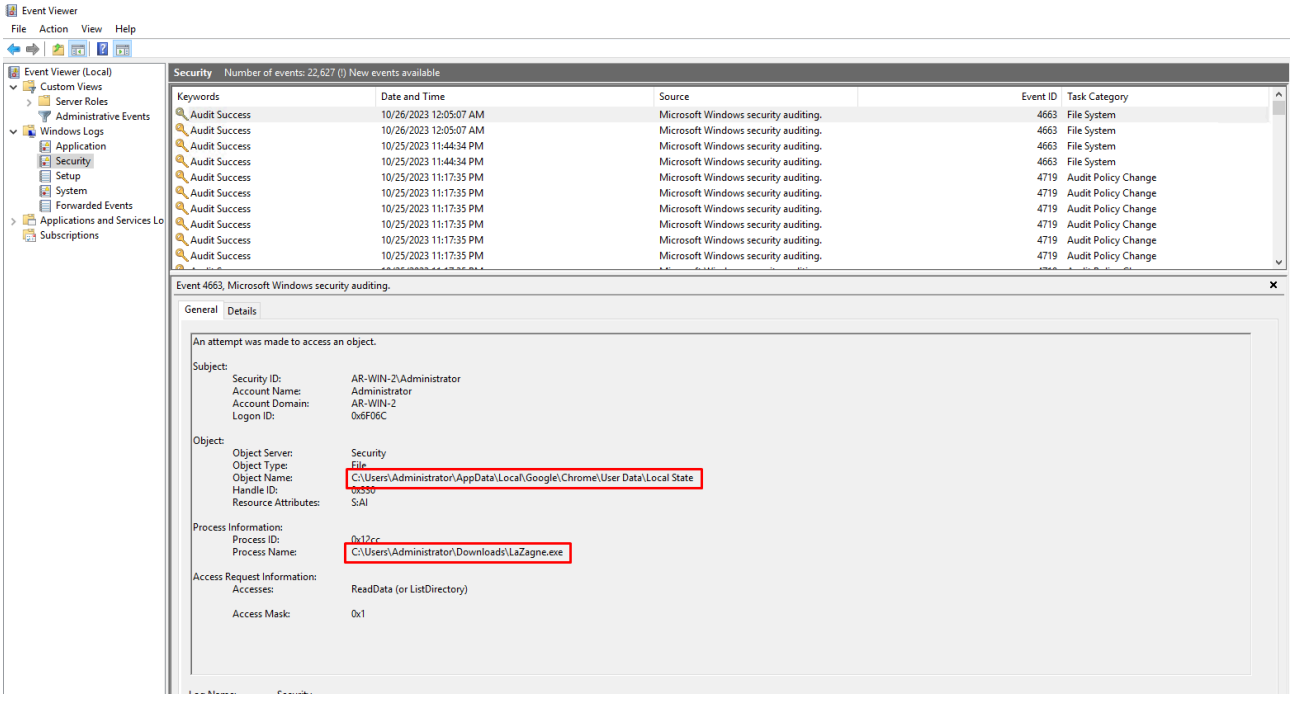
## Ensuring Visibility

After enabling Windows event logs, if you execute Lazagne again, you can see the event logs that indicate the execution of Lazagne with the "browsers" parameter.



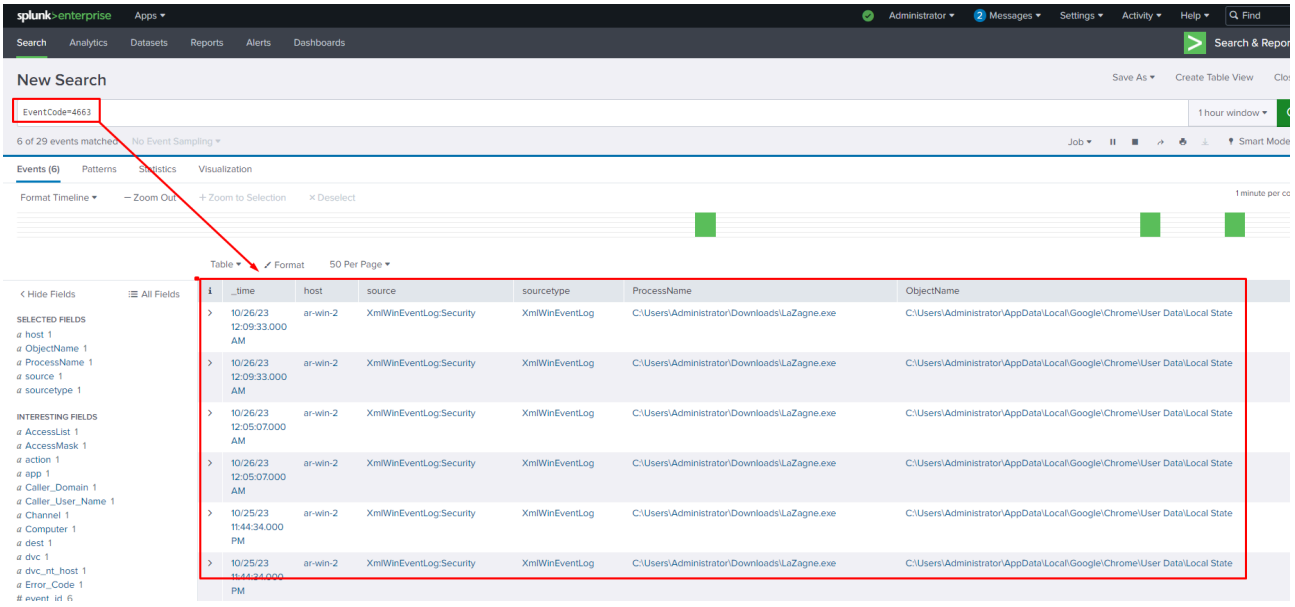

Lazagne Process Creation Event view in Event Viewer

We can also see the File Access Event Logs in Event Viewer → Windows Logs → Security logs by following Windows 4663 Events (**An attempt was made to access an object**)




File Access Event in Event Viewer

These events can also be forwarded to your SIEM dashboard so you can build appropriate alerts for these behaviours.




Splunk Dashboard filter for File Access Events

## Detection Rules for the Win

By analyzing event logs, we can create a [Sigma rule] that can detect any unauthorized attempt to execute Lazagne for dumping browser credentials. We can use these rules to detect malicious or unauthorized access to browser credentials.

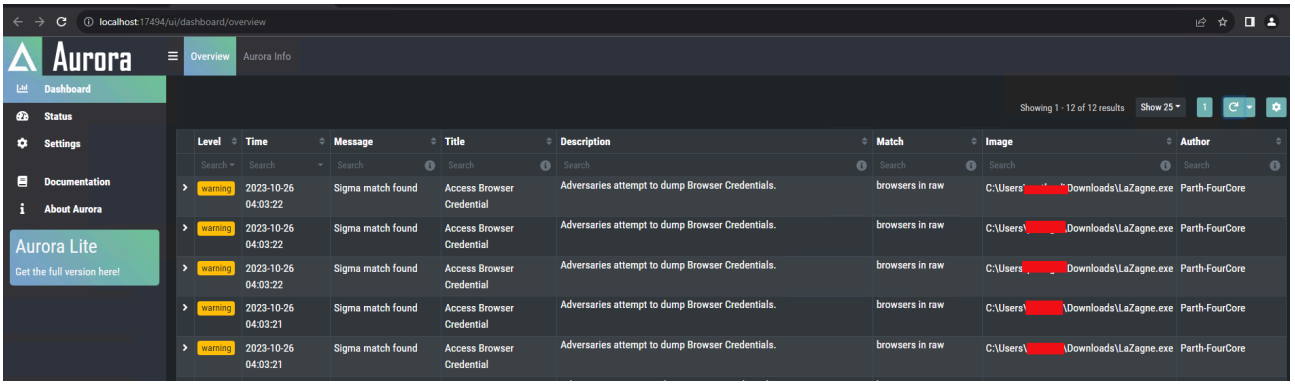
### Command Line Detection

Command line detections are based on default and known command line patterns used by threat actors during the execution of the binary. For any unmodified tool, we can detect its presence either via its hash or by the known command line for this tool. The Sigma rule will detect the presence of the keyword "browser" in the command line along with other keywords commonly used by Lazagne to extract browser credentials.

```
 1title: Access Browser Credential
2description: Adversaries may search for common password storage locations to obtain user credentials.
3id: 198c1a5c-72cc-11ee-b962-0242ac120002
4status: test
5author: Parth-FourCore
6date: 2023/10/21
7tags:
8    - attack.t1003
9    - attack.credential_access
10logsource:
11    product: windows
12detection:
13    keywords:
14        - 'browsers'
```

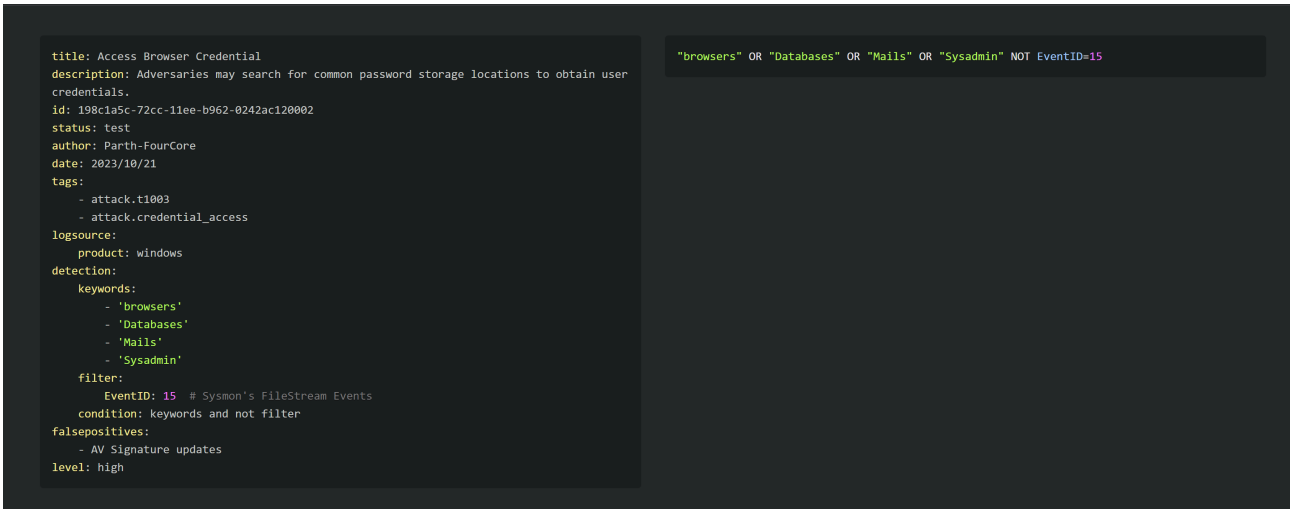
```
        - 'Databases'
        - 'Mails'
        - 'Sysadmin'
    filter:
        EventID: 15  # Sysmon's FileStream Events
    condition: keywords and not filter
falsepositives:
    - AV Signature updates
    - Files with Browsers in their filename
level: high
```

In order to test the created Sigma rule, we can utilise Aurora. Aurora is a lightweight and customisable EDR that is based on Sigma rules and can be quickly set up to test your rules.




Aurora Dashboard

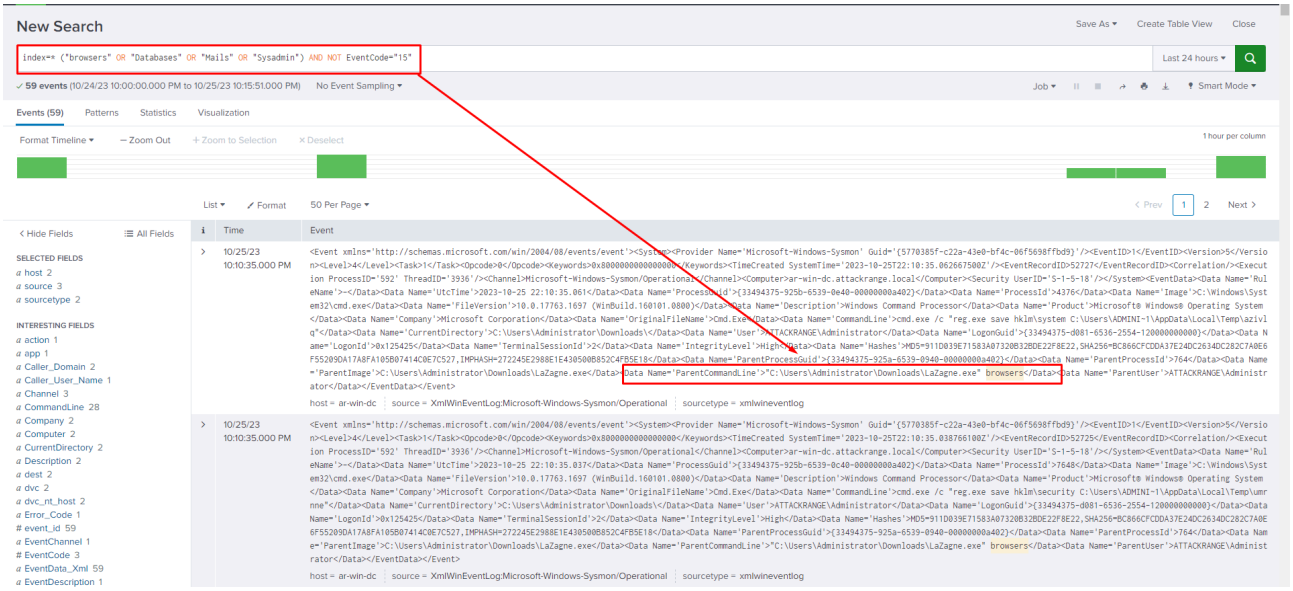
We can use the same Sigma rule to convert into SIEM, EDR, XDR, and data lake query formats to search related event logs and set alerts.



```
title: Access Browser Credential
description: Adversaries may search for common password storage locations to obtain user
credentials.
id: 198c1a5c-72cc-11ee-b962-0242ac120002
status: test
author: Parth-FourCore
date: 2023/10/21
tags:
    - attack.t1003
    - attack.credential_access
logsource:
    product: windows
detection:
    keywords:
        - 'browsers'
        - 'Databases'
        - 'Mails'
        - 'Sysadmin'
    filter:
        EventID: 15  # Sysmon's FileStream Events
    condition: keywords and not filter
falsepositives:
    - AV Signature updates
level: high
```

```
"browsers" OR "Databases" OR "Mails" OR "Sysadmin" NOT EventID=15
```

Let us use this search query in the Splunk dashboard



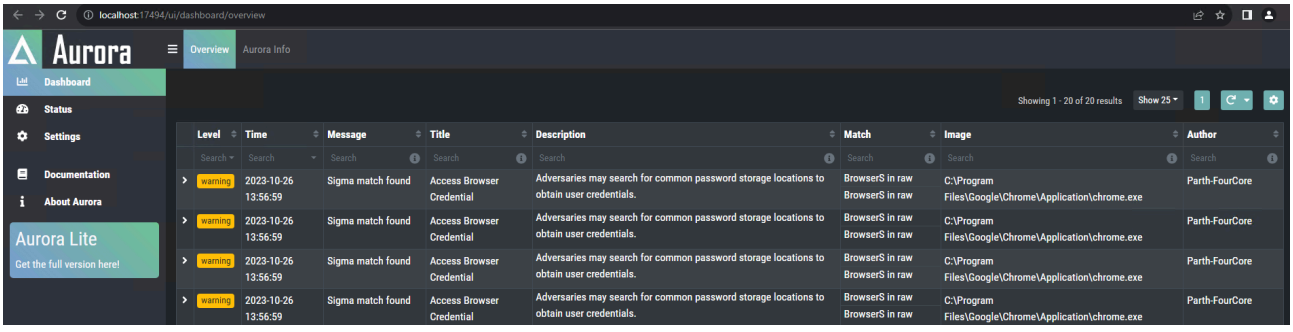

Splunk Dashboard

## Concerns with Command Line Detections

Command Line detections are very useful for closed source detections or detecting script kiddies which directly use tools without understanding the methodologies.

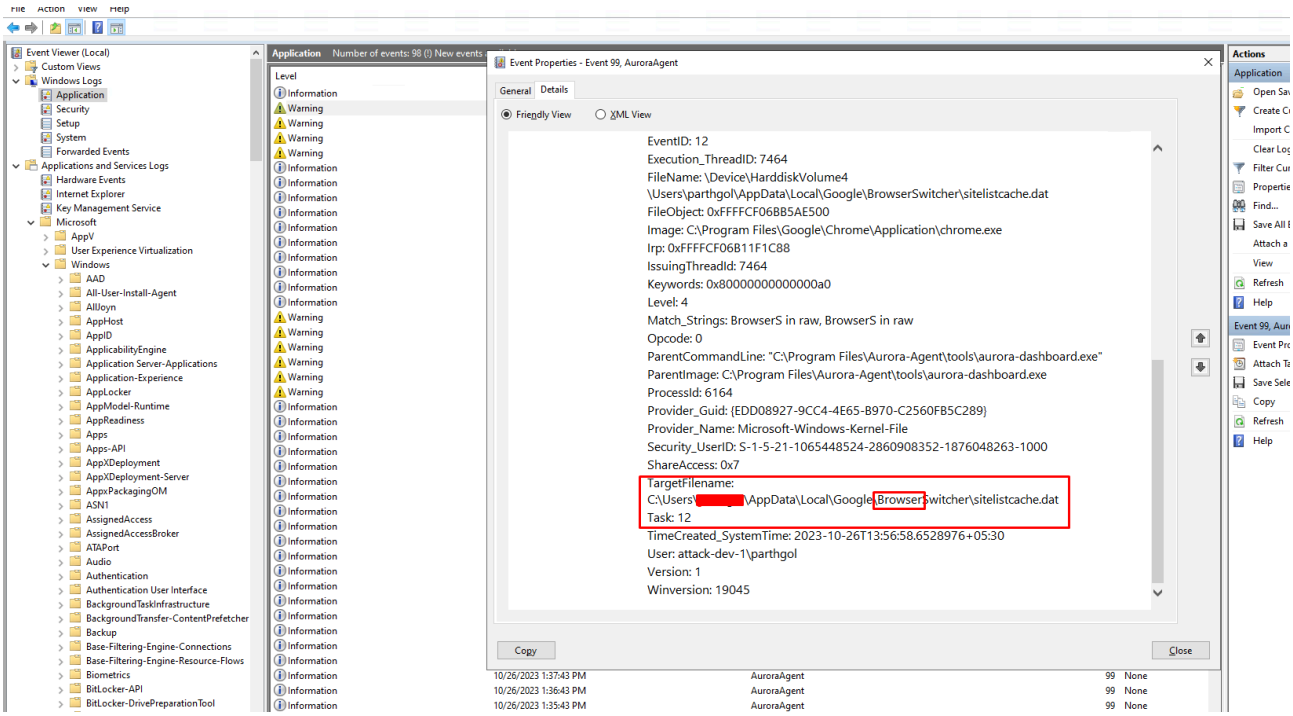
On the other hand, these rules are brittle for open-source toolkits as these tools can be easily modified, and their command line parameters can be spoofed or modified. Since the rule specifically targets binaries with certain keywords, it won't detect any changes to default tools.

Also, command line tools can have lots of false positives because the same parameters or keywords can be present in other non-malicious binaries




False positive in Detection Rule

This alert is triggered by the chrome.exe binary, which is a legitimate browser. However, the keyword "Browsers" in Event triggers the alert.




False Positive Event for Browser Credentials

---

## Detect Behaviours not Tools

We can create a different sigma rule which, rather than focusing on command line parameters, focuses on the file access events by the unknown process to alert for malicious behaviours. It is essential to have all the browser paths mentioned in the Sigma rule so that we can monitor access events for all available browsers on the host machine.

```
 1- '\cookies.sqlite'
2- 'release\key3.db' # Firefox
3- 'release\key4.db' # Firefox
4- 'release\logins.json' # Firefox
5- '\Appdata\Local\Chrome\User Data\Default\Login Data' # Crome
6- '\AppData\Local\Google\Chrome\User Data\Default\Network\Cookies' # googel crome
7- '\AppData\Local\Google\Chrome\User Data\Local State'
8- '\Appdata\Local\7Star\7Star\User Data' # 7Star
9- '\Appdata\Local\Amigo\User Data' # amigo
10- '\Appdata\Local\BraveSoftware\Brave-Browser\User Data' # brave
11- '\Appdata\Local\CentBrowser\User Data' # centbrowser
12- '\Appdata\Local\Chedot\User Data' # chedot
13- '\Appdata\Local\Google\Chrome SxS\User Data' # chrome canary
14- '\Appdata\Local\Chromium\User Data' # chromium
15- '\Appdata\Local\Microsoft\Edge\User Data' # chromium edge
16- '\Appdata\Local\CocCoc\Browser\User Data' # coccoc
17- '\Appdata\Local\Comodo\Dragon\User Data' # Comodo IceDragon is Firefox-based
```

```
18- '\Appdata\Local\Elements Browser\User Data' # elements browser
19- '\Appdata\Local\Epic Privacy Browser\User Data' # epic privacy browser
20- '\Appdata\Local\Kometa\User Data' # kometa
21- '\Appdata\Opera Software\Opera Stable' # opera
22- '\Appdata\Local\Orbitum\User Data' # orbitum
23- '\Appdata\Local\Sputnik\Sputnik\User Data' # sputnik
24- '\Appdata\Local\Torch\User Data' # torch
25- '\Appdata\Local\uCozMedia\Uran\User Data' # uran
26- '\Appdata\Local\Vivaldi\User Data' # vivaldi
27- '\Appdata\Local\Yandex\YandexBrowser\User Data' # yandexBrowser
28- '\Appdata\Local\Mozilla\Firefox' # firefox
29- '\Appdata\Local\NETGATE Technologies\BlackHawk' # blackHawk
30- '\Appdata\Local\8pecxstudios\Cyberfox' # cyberfox
31- '\Appdata\Local\Comodo\IceDragon' # comodo IceDragon
32- '\Appdata\Local\K-Meleon' # k-Meleon
33- '\Appdata\Local\Mozilla\icecat' # icecat
34- '\Appdata\Local\UCBrowser' #  UCbrowser
```

### Weeding out False Positives

Since these files and paths are not only accessed by unauthorized tools and processes but also used by antivirus software, legitimate binaries, 3rd party backup software, and other authorized tools in your environment, It is crucial to add appropriate filters to the Sigma rule to prevent false positives. The following list is what we observed in our test environment, which are false positives. Auditing the rules in your production environment is crucial to eliminate false positives.

| **Windows defender** | **C:\ProgramData\Microsoft\Windows Defender\MsMpEng.exe** |
|---|---|
| Windows Installer | C:\Windows\System32\msiexec.exe |
| Browser Files | chrome.exe, edge.exe |

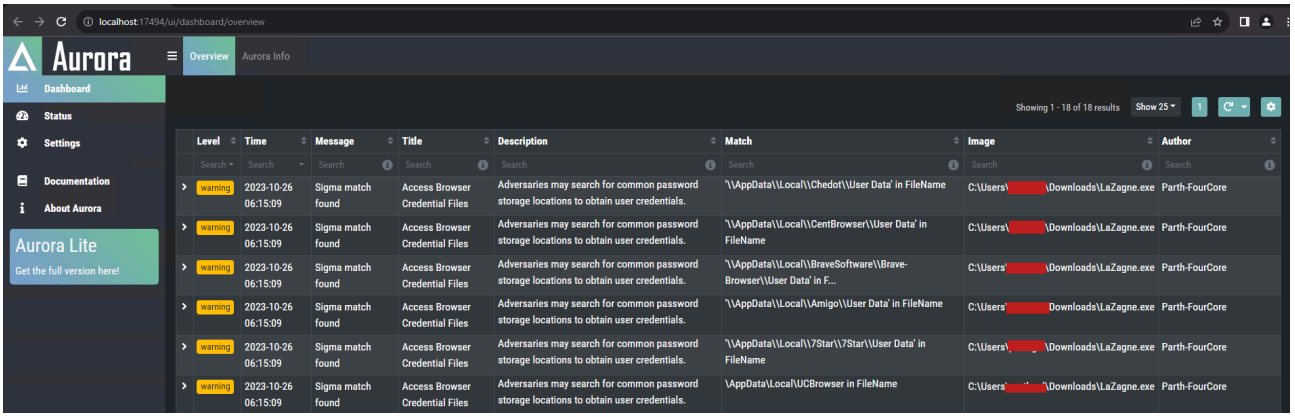
Here, adding all file and folder paths with a filter parameter will help avoid false positives.

```
 1title: Access Browser Credential Files
2description: Adversaries may search for common password storage locations to obtain user credentials.
3id: 198c1a5c-72cc-11ee-b962-0242ac120002
4status: experimental
5author: Parth-FourCore
6date: 2023/10/24
7tags:
8  - attack.t1003
9  - attack.credential_access
10logsource:
11  category: file_access
```

```
product: windows
detection:
  selection_all:
    ObjectName|contains:
      - '\cookies.sqlite'
      - 'release\key3.db' # Firefox
      - 'release\key4.db' # Firefox
      - 'release\logins.json' # Firefox
      - '\Appdata\Local\Chrome\User Data\Default\Login Data' # Crome
      - '\AppData\Local\Google\Chrome\User Data\Default\Network\Cookies' # googel crome
      - '\AppData\Local\Google\Chrome\User Data\Local State'
      - '\Appdata\Local\7Star\7Star\User Data' # 7Star
      - '\Appdata\Local\Amigo\User Data' # amigo
      - '\Appdata\Local\BraveSoftware\Brave-Browser\User Data' # brave
      - '\Appdata\Local\CentBrowser\User Data' # centbrowser
      - '\Appdata\Local\Chedot\User Data' # chedot
      - '\Appdata\Local\Google\Chrome SxS\User Data' # chrome canary
      - '\Appdata\Local\Chromium\User Data' # chromium
      - '\Appdata\Local\Microsoft\Edge\User Data' # chromium edge
      - '\Appdata\Local\CocCoc\Browser\User Data' # coccoc
      - '\Appdata\Local\Comodo\Dragon\User Data' # Comodo IceDragon is Firefox-based
      - '\Appdata\Local\Elements Browser\User Data' # elements browser
      - '\Appdata\Local\Epic Privacy Browser\User Data' # epic privacy browser
      - '\Appdata\Local\Kometa\User Data' # kometa
      - '\Appdata\Opera Software\Opera Stable' # opera
      - '\Appdata\Local\Orbitum\User Data' # orbitum
      - '\Appdata\Local\Sputnik\Sputnik\User Data' # sputnik
      - '\Appdata\Local\Torch\User Data' # torch
      - '\Appdata\Local\uCozMedia\Uran\User Data' # uran
      - '\Appdata\Local\Vivaldi\User Data' # vivaldi
      - '\Appdata\Local\Yandex\YandexBrowser\User Data' # yandexBrowser
      - '\Appdata\Local\Mozilla\Firefox' # firefox
      - '\Appdata\Local\NETGATE Technologies\BlackHawk' # blackHawk
      - '\Appdata\Local\8pecxstudios\Cyberfox' # cyberfox
      - '\Appdata\Local\Comodo\IceDragon' # comodo IceDragon
      - '\Appdata\Local\K-Meleon' # k-Meleon
      - '\Appdata\Local\Mozilla\icecat' # icecat
      - '\Appdata\Local\UCBrowser' #  UCbrowser
  filter_main_system:
    Image: System
    ParentImage: Idle
  filter_main_generic:
    Image|startswith:
      - 'C:\Program Files\'
      - 'C:\Program Files (x86)\'
      - 'C:\WINDOWS\system32\'
      - 'C:\WINDOWS\SysWOW64\'
  filter_optional_defender:
    Image|startswith: 'C:\ProgramData\Microsoft\Windows Defender\'
    Image|endswith:
      - '\MpCopyAccelerator.exe'
      - '\MsMpEng.exe'
```

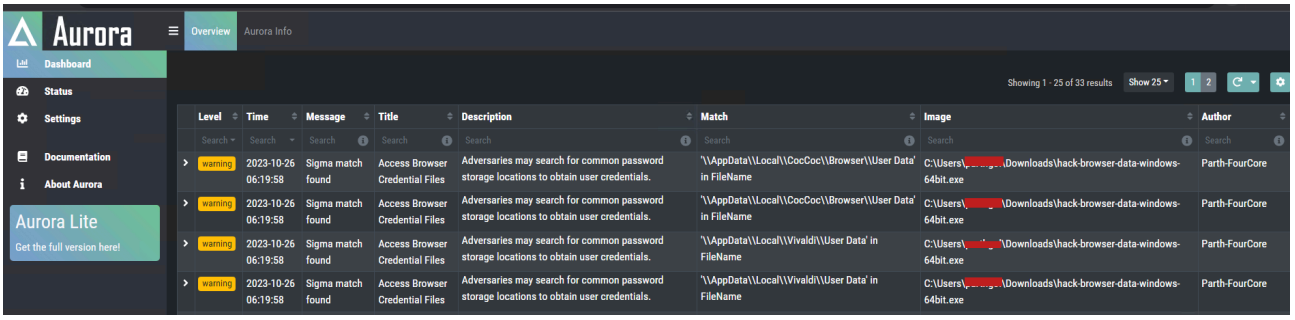
```
64  filter_optional_msiexec:
65    ParentImage: 'C:\Windows\System32\msiexec.exe'
66  condition: selection_all and not 1 of filter_main_* and not 1 of filter_optional_*
67falsepositives:
68  - Antivirus, Anti-Spyware, Anti-Malware Software
69  - Backup software
70  - Legitimate software
71level: high
```

Let's execute the lazagne and HackBrowserData tools with the Aurora agent to verify the new Sigma rule.




Aurora Dashboard

We will receive an alert when HackBrowserData attempts to access browser credential files as well.

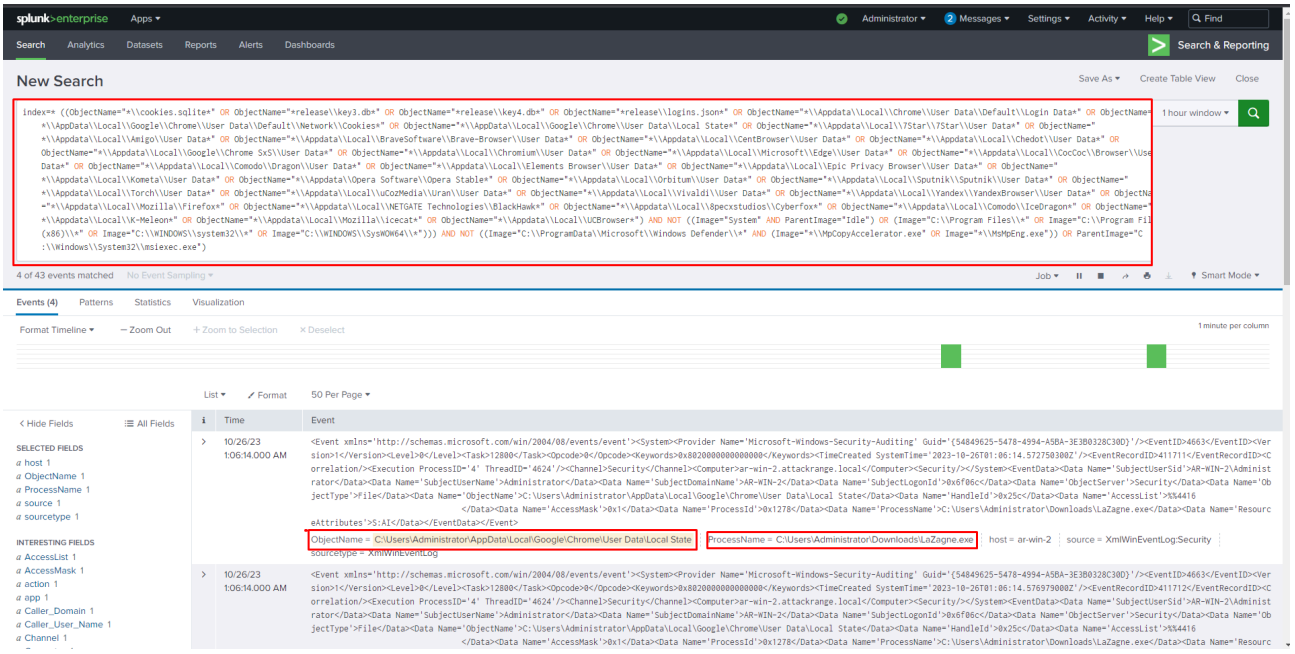



Aurora Dashboard

These rules are now ready for use in our environment, albeit with a clause. This rule will not get triggered in case of a process injection attack; however, we will discuss that in a future blog. Identifying false positives and updating your rules accordingly is a continuous process.

## Setting up alerts

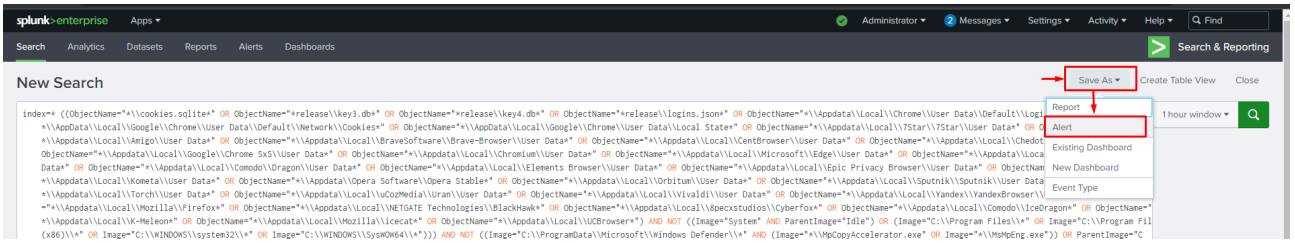
Let's convert the Sigma rule to a Splunk query and use it to search in the Splunk dashboard.




Splunk Dashboard

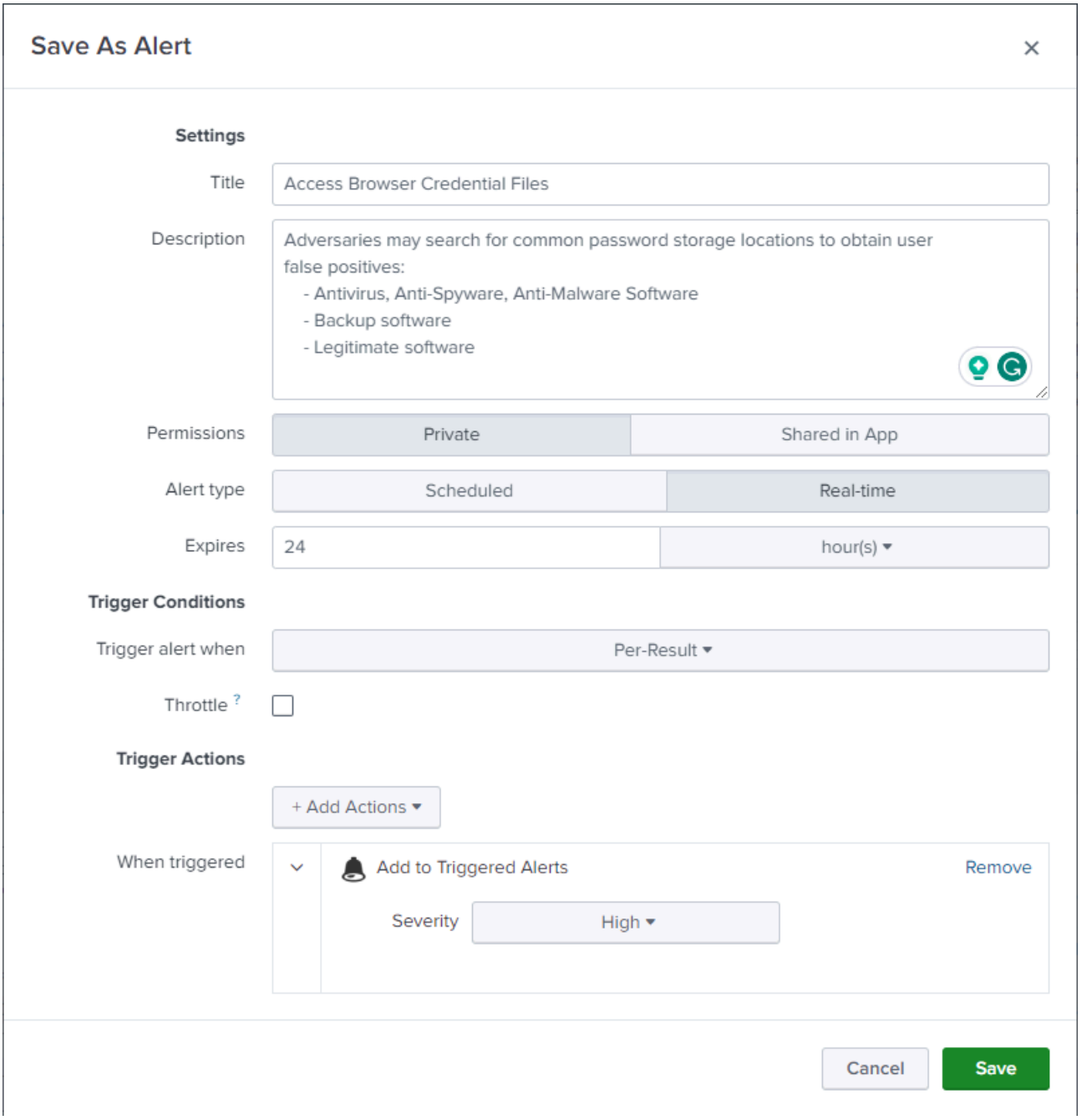
We can see the Splunk showcases logs for Lazagne, trying to extract stored login information from Google Chrome's saved password file.

We can alert on this behaviour and prevent future threats by converting the query to an alert by simply clicking on "Save As" and selecting "Alert".



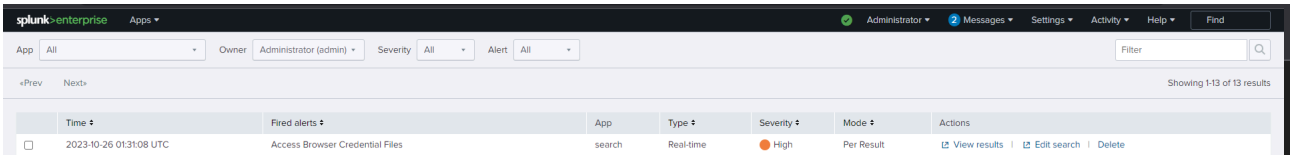

Adding New Aleart in Splunk

In the "Save As" alert menu, please ensure that you fill in all the necessary details for the alert.



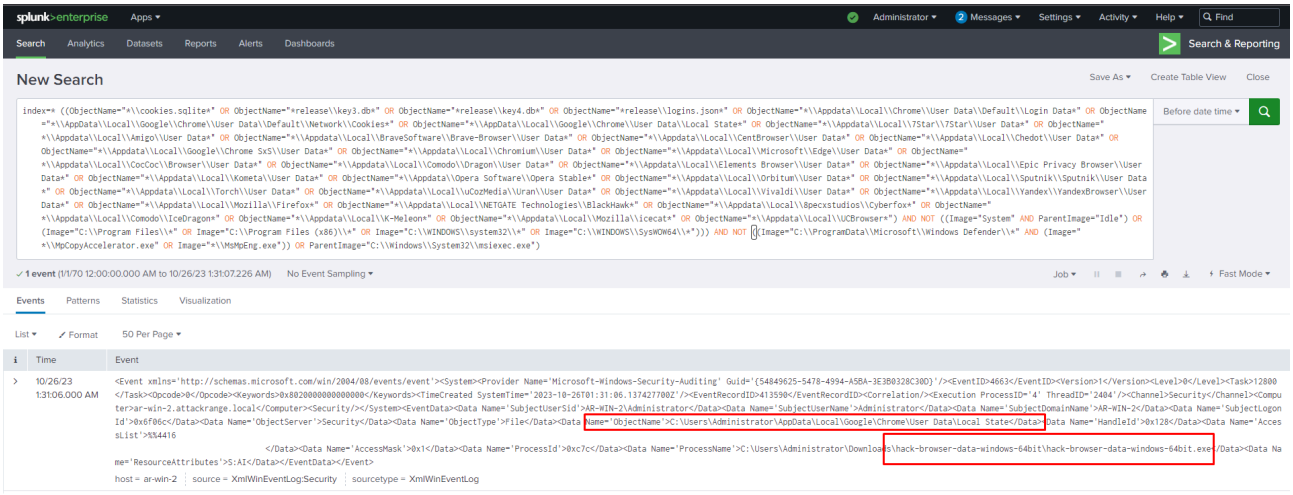

Alert Menu

Click on the Save button to save the alert. Now, let's try triggering the alert with different tools to ensure that the alert we created using a Sigma rule works with any browser credential extraction tool. Once configured, we can see that the alert is also triggered with different hacking tools.




Triggered Alert

When you click on "View Result", you will be able to locate the event that triggered this alert.

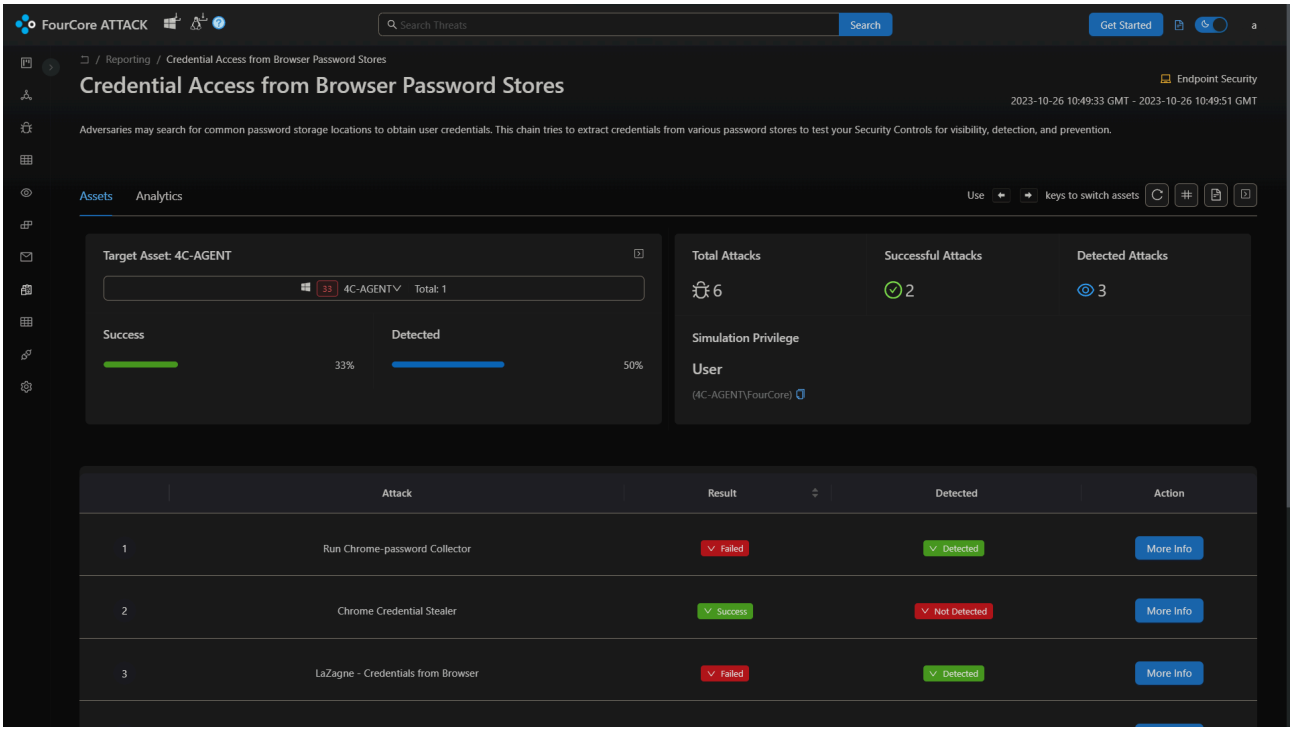



Event which Trigged Alert

By specifying file names and file paths in our Sigma rule, we can detect any unauthorised access to valuable files such as usernames and passwords stored in the browser. Additionally, we can identify any unauthorised execution of hacker tools which try to obtain browser credentials.

## Browser Credential Access with FourCore ATTACK

> The FourCore ATTACK platform can emulate the different types of browser-based credential access techniques, such as via LaZagne, via using Powershell or by accessing the files directly. These variants can be hunted using the Sigma rules shared in this post.



Learn about **writing your own Sigma rule** via this deep dive into working with Sigma.

You can also read more about using **Windows Event Log IDs for threat-hunting** here.

## References


1. Browser Password Managers
2. Redline Stealer
3. Browser Credential Harvesting MITRE
4. Nirsoft WebBrowserPassView
5. LaZagne open-source password stealer
6. Process Auditing Event ID 4663
7. Process Auditing Event ID 4688
8. Sigma Rules Github
9. Learn More about Splunk


---