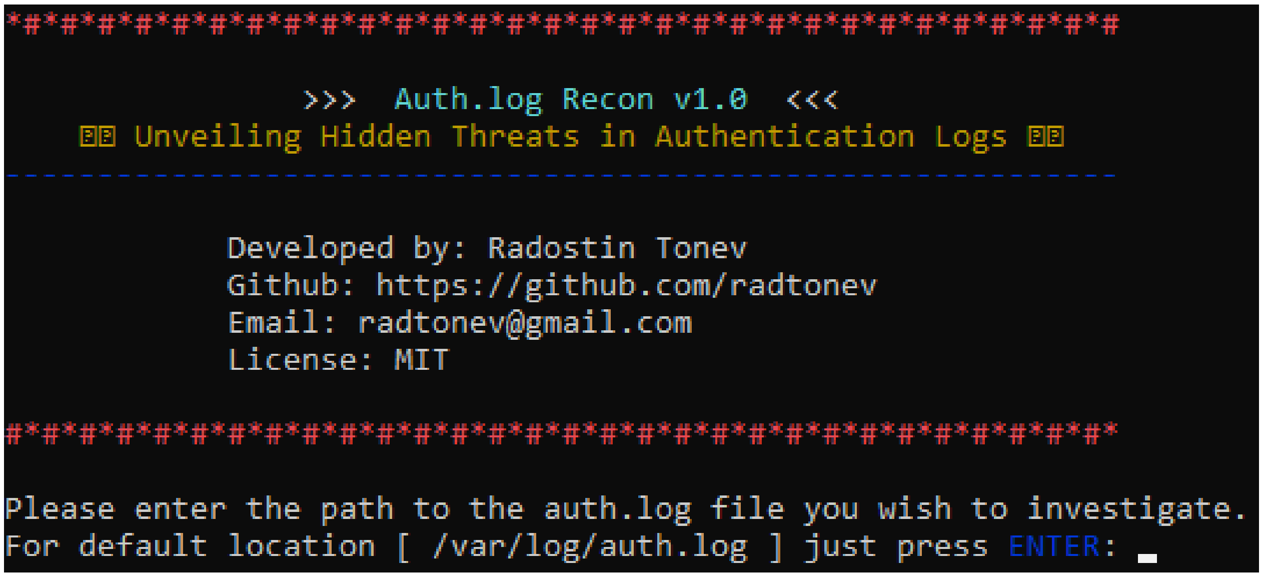
Auth.log Recon v1.0 - User Guide



```
*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#

                >>>  Auth.log Recon v1.0  <<<
    ⍰⍰ Unveiling Hidden Threats in Authentication Logs ⍰⍰
------------------------------------------------------------

            Developed by: Radostin Tonev
            Github: https://github.com/radtonev
            Email: radtonev@gmail.com
            License: MIT

#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*

Please enter the path to the auth.log file you wish to investigate.
For default location [ /var/log/auth.log ] just press ENTER: _
```

Overview

Auth.log Recon v1.0 is a Python-based utility designed to thoroughly analyze authentication logs, specifically `auth.log` file in Linux, and extract critical security-related information. It empowers security professionals and system administrators to efficiently identify potential security breaches, suspicious activities, and system anomalies that might be hidden within the extensive log data. This tool streamlines the process of log analysis, providing clear insights into user activities, login patterns, and potential security threats.

Installation

1. **Python Requirement:** Ensure that your system has Python 3.6 or a more recent version installed. You can verify this by opening a terminal or command prompt and typing `python3 --version`. If Python is not installed, you'll need to download and install it from the official Python website.
2. **Script Acquisition:** Obtain the Python script (e.g., `auth_log_analyzer.py`). This typically involves saving the script to a directory on your computer where you can easily access it.
3. **Dependency Check:** This script relies on standard Python libraries, so there are no additional installations required. These libraries are included with most Python distributions.

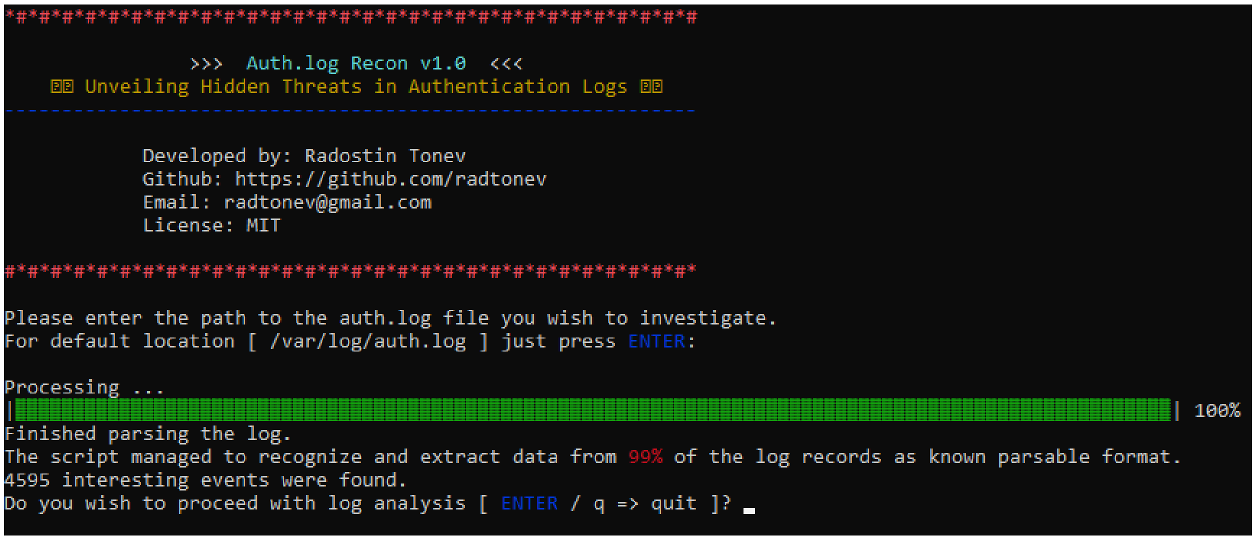
Usage

1. **Script Execution:** Open your terminal or command prompt and navigate to the directory where you saved the `auth_log_analyzer.py` file. Execute the script by typing `python3 auth_log_analyzer.py` (or `python auth_log_analyzer.py` if Python 3 is your default Python version) and press Enter.

2. **Log File Path Input:**
   The script will prompt you to provide the location of the `auth.log` file that you intend to analyze. You have two options: you can either enter the complete path to the file (e.g., `/var/log/auth.log`) or, if you want to use the default location, simply press the Enter key. The default path is often `/var/log/auth.log`, but this can vary slightly depending on your system.

3. **Log File Processing:**
   The script will begin processing the log file. During this process, it extracts relevant events and displays a progress bar to indicate the analysis's progress. This can take some time, depending on the size of the log file.

   If the script encounters a situation where it cannot correctly interpret a significant portion of the log records (meaning it doesn't recognize the log format), it will display a *warning message*. You will then be asked to confirm whether you wish to continue with the analysis. This warning is important because the accuracy of the results might be affected if a large number of log entries are not parsed correctly.

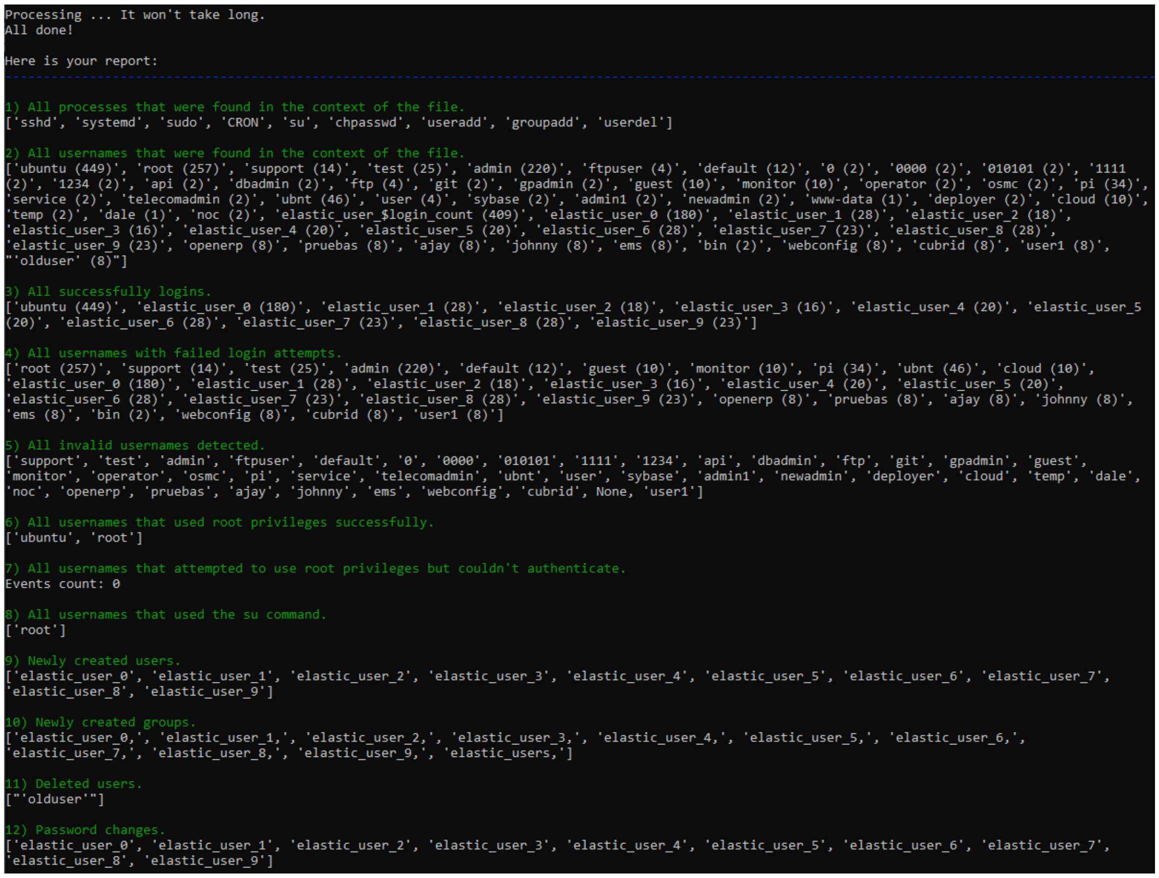


```
*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#

                >>>  Auth.log Recon v1.0  <<<
    ▯▯ Unveiling Hidden Threats in Authentication Logs ▯▯
------------------------------------------------------------

            Developed by: Radostin Tonev
            Github: https://github.com/radtonev
            Email: radtonev@gmail.com
            License: MIT

#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*

Please enter the path to the auth.log file you wish to investigate.
For default location [ /var/log/auth.log ] just press ENTER:

Processing ...
|████████████████████████████████████████████████████████████████████████████████████████████████████| 100%
Finished parsing the log.
The script managed to recognize and extract data from 99% of the log records as known parsable format.
4595 interesting events were found.
Do you wish to proceed with log analysis [ ENTER / q => quit ]? _
```

4. **Analysis Report Generation:**
   Upon completion of the log file processing, the script will generate a detailed report. This report summarizes the key information that has been extracted from the log file, presenting it in a structured and readable format. The report typically includes the following categories of information:

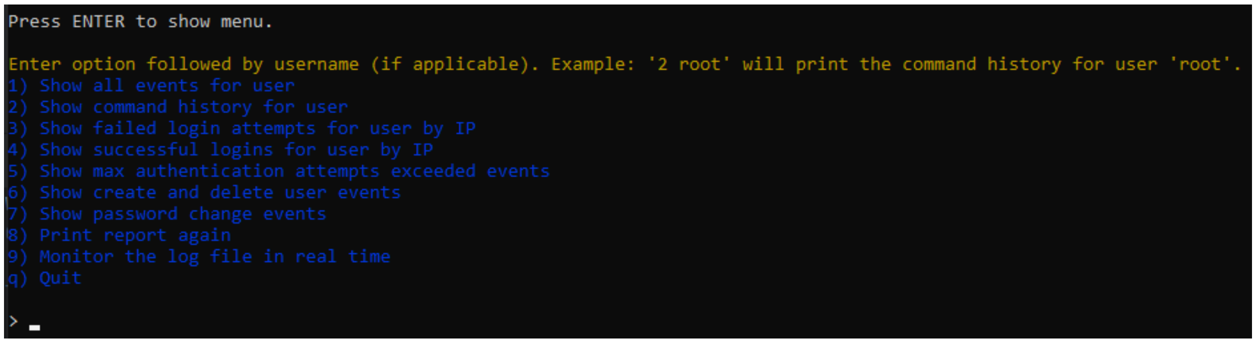
- **Processes:** A list of all the processes that were found to be recorded within the log file. This can help in understanding which applications or services were active on the system.
- **Usernames:** A comprehensive list of all usernames that are referenced in the log file. This is crucial for identifying user activity and potential account-related issues.

- **Successful Logins:** A summary of all successful login events, which can be useful for tracking user access to the system.
- **Failed Logins:** A summary of all failed login attempts, which are critical for detecting potential brute-force attacks or unauthorized access attempts.
- **Invalid Usernames:** A list of usernames that were used in login attempts but were found to be invalid. This can indicate attempts to guess valid account names.
- **`sudo` Command Usage:** Information about the use of the `sudo` command, including which users executed commands with elevated privileges.
- **`sudo` Command Failures:** Details about attempts to use the `sudo` command that were unsuccessful due to authentication failures.
- **`su` Command Usage:** Information about the usage of the `su` command, which is another command used to switch user accounts.
- **User Creation Events:** Records of when new user accounts were created on the system.
- **User Deletion Events:** Records of when user accounts were deleted from the system.
- **Group Creation Events:** Records of when new user groups were created.
- **Password Change Events:** Records of user password changes.



```
Processing ... It won't take long.
All done!

Here is your report:
----------------------------------------------------------------------------------------------------------------------------

1) All processes that were found in the context of the file.
['sshd', 'systemd', 'sudo', 'CRON', 'su', 'chpasswd', 'useradd', 'groupadd', 'userdel']

2) All usernames that were found in the context of the file.
['ubuntu (449)', 'root (257)', 'support (14)', 'test (25)', 'admin (220)', 'ftpuser (4)', 'default (12)', '0 (2)', '0000 (2)', '010101 (2)', '1111
(2)', '1234 (2)', 'api (2)', 'dbadmin (2)', 'ftp (4)', 'git (2)', 'gpadmin (2)', 'guest (10)', 'monitor (10)', 'operator (2)', 'osmc (2)', 'pi (34)',
'service (2)', 'telecomadmin (2)', 'ubnt (46)', 'user (4)', 'sybase (2)', 'admin1 (2)', 'newadmin (2)', 'www-data (1)', 'deployer (2)', 'cloud (10)',
'temp (2)', 'dale (1)', 'noc (2)', 'elastic_user_$login_count (409)', 'elastic_user_0 (180)', 'elastic_user_1 (28)', 'elastic_user_2 (18)',
'elastic_user_3 (16)', 'elastic_user_4 (20)', 'elastic_user_5 (20)', 'elastic_user_6 (28)', 'elastic_user_7 (23)', 'elastic_user_8 (28)',
'elastic_user_9 (23)', 'openerp (8)', 'pruebas (8)', 'ajay (8)', 'johnny (8)', 'ems (8)', 'bin (2)', 'webconfig (8)', 'cubrid (8)', 'user1 (8)',
"'olduser' (8)"]

3) All successfully logins.
['ubuntu (449)', 'elastic_user_0 (180)', 'elastic_user_1 (28)', 'elastic_user_2 (18)', 'elastic_user_3 (16)', 'elastic_user_4 (20)', 'elastic_user_5
(20)', 'elastic_user_6 (28)', 'elastic_user_7 (23)', 'elastic_user_8 (28)', 'elastic_user_9 (23)']

4) All usernames with failed login attempts.
['root (257)', 'support (14)', 'test (25)', 'admin (220)', 'default (12)', 'guest (10)', 'monitor (10)', 'pi (34)', 'ubnt (46)', 'cloud (10)',
'elastic_user_0 (180)', 'elastic_user_1 (28)', 'elastic_user_2 (18)', 'elastic_user_3 (16)', 'elastic_user_4 (20)', 'elastic_user_5 (20)',
'elastic_user_6 (28)', 'elastic_user_7 (23)', 'elastic_user_8 (28)', 'elastic_user_9 (23)', 'openerp (8)', 'pruebas (8)', 'ajay (8)', 'johnny (8)',
'ems (8)', 'bin (2)', 'webconfig (8)', 'cubrid (8)', 'user1 (8)']

5) All invalid usernames detected.
['support', 'test', 'admin', 'ftpuser', 'default', '0', '0000', '010101', '1111', '1234', 'api', 'dbadmin', 'ftp', 'git', 'gpadmin', 'guest',
'monitor', 'operator', 'osmc', 'pi', 'service', 'telecomadmin', 'ubnt', 'user', 'sybase', 'admin1', 'newadmin', 'deployer', 'cloud', 'temp', 'dale',
'noc', 'openerp', 'pruebas', 'ajay', 'johnny', 'ems', 'webconfig', 'cubrid', None, 'user1']

6) All usernames that used root privileges successfully.
['ubuntu', 'root']

7) All usernames that attempted to use root privileges but couldn't authenticate.
Events count: 0

8) All usernames that used the su command.
['root']

9) Newly created users.
['elastic_user_0', 'elastic_user_1', 'elastic_user_2', 'elastic_user_3', 'elastic_user_4', 'elastic_user_5', 'elastic_user_6', 'elastic_user_7',
'elastic_user_8', 'elastic_user_9']

10) Newly created groups.
['elastic_user_0,', 'elastic_user_1,', 'elastic_user_2,', 'elastic_user_3,', 'elastic_user_4,', 'elastic_user_5,', 'elastic_user_6,',
'elastic_user_7,', 'elastic_user_8,', 'elastic_user_9,', 'elastic_users,']

11) Deleted users.
["'olduser'"]

12) Password changes.
['elastic_user_0', 'elastic_user_1', 'elastic_user_2', 'elastic_user_3', 'elastic_user_4', 'elastic_user_5', 'elastic_user_6', 'elastic_user_7',
'elastic_user_8', 'elastic_user_9']
```

5. **Interactive Menu Interface:**
   After the analysis report is displayed, the script presents an interactive menu. This menu allows you to perform more specific queries and investigations based on the data extracted from the log file. This interactive feature provides flexibility in how you analyze the log data.

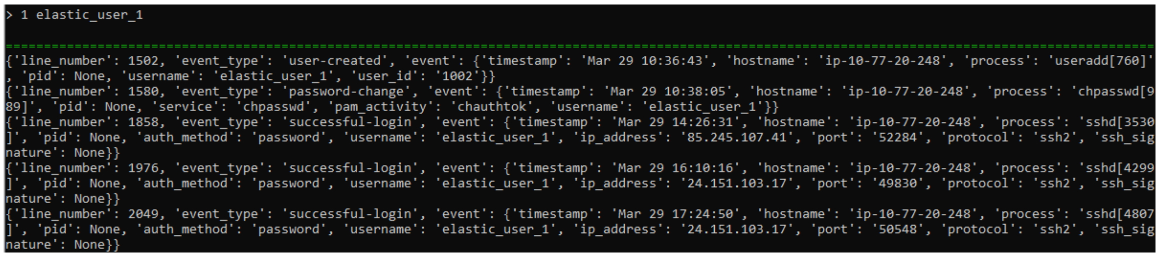


```
Press ENTER to show menu.

Enter option followed by username (if applicable). Example: '2 root' will print the command history for user 'root'.
1) Show all events for user
2) Show command history for user
3) Show failed login attempts for user by IP
4) Show successful logins for user by IP
5) Show max authentication attempts exceeded events
6) Show create and delete user events
7) Show password change events
8) Print report again
9) Monitor the log file in real time
q) Quit

> _
```

Interactive Menu Options

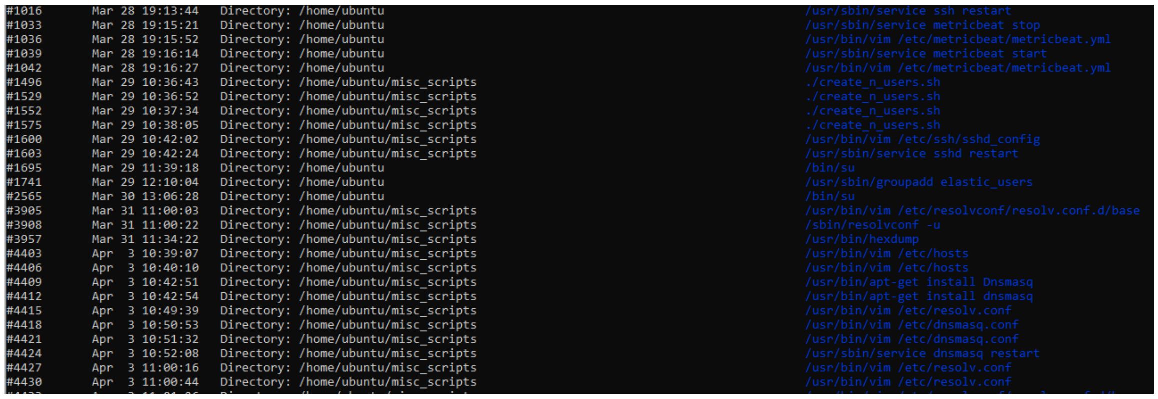
The interactive menu offers a range of options to help you delve deeper into the log data:

- **1) Show All Events for User:** This option allows you to display all recorded events that are associated with a specific username. This provides a chronological view of a user's activities as captured in the log in a raw JSON format.
  Example: If you want to see all the log entries related to the user "elastic_user_1" you would enter `1 elastic_user_1` and press Enter.



```
> 1 elastic_user_1

====================================================================================================================================
{'line_number': 1502, 'event_type': 'user-created', 'event': {'timestamp': 'Mar 29 10:36:43', 'hostname': 'ip-10-77-20-248', 'process': 'useradd[760]'
, 'pid': None, 'username': 'elastic_user_1', 'user_id': '1002'}}
{'line_number': 1580, 'event_type': 'password-change', 'event': {'timestamp': 'Mar 29 10:38:05', 'hostname': 'ip-10-77-20-248', 'process': 'chpasswd[9
89]', 'pid': None, 'service': 'chpasswd', 'pam_activity': 'chauthtok', 'username': 'elastic_user_1'}}
{'line_number': 1858, 'event_type': 'successful-login', 'event': {'timestamp': 'Mar 29 14:26:31', 'hostname': 'ip-10-77-20-248', 'process': 'sshd[3530
]', 'pid': None, 'auth_method': 'password', 'username': 'elastic_user_1', 'ip_address': '85.245.107.41', 'port': '52284', 'protocol': 'ssh2', 'ssh_sig
nature': None}}
{'line_number': 1976, 'event_type': 'successful-login', 'event': {'timestamp': 'Mar 29 16:10:16', 'hostname': 'ip-10-77-20-248', 'process': 'sshd[4299
]', 'pid': None, 'auth_method': 'password', 'username': 'elastic_user_1', 'ip_address': '24.151.103.17', 'port': '49830', 'protocol': 'ssh2', 'ssh_sig
nature': None}}
{'line_number': 2049, 'event_type': 'successful-login', 'event': {'timestamp': 'Mar 29 17:24:50', 'hostname': 'ip-10-77-20-248', 'process': 'sshd[4807
]', 'pid': None, 'auth_method': 'password', 'username': 'elastic_user_1', 'ip_address': '24.151.103.17', 'port': '50548', 'protocol': 'ssh2', 'ssh_sig
nature': None}}
```

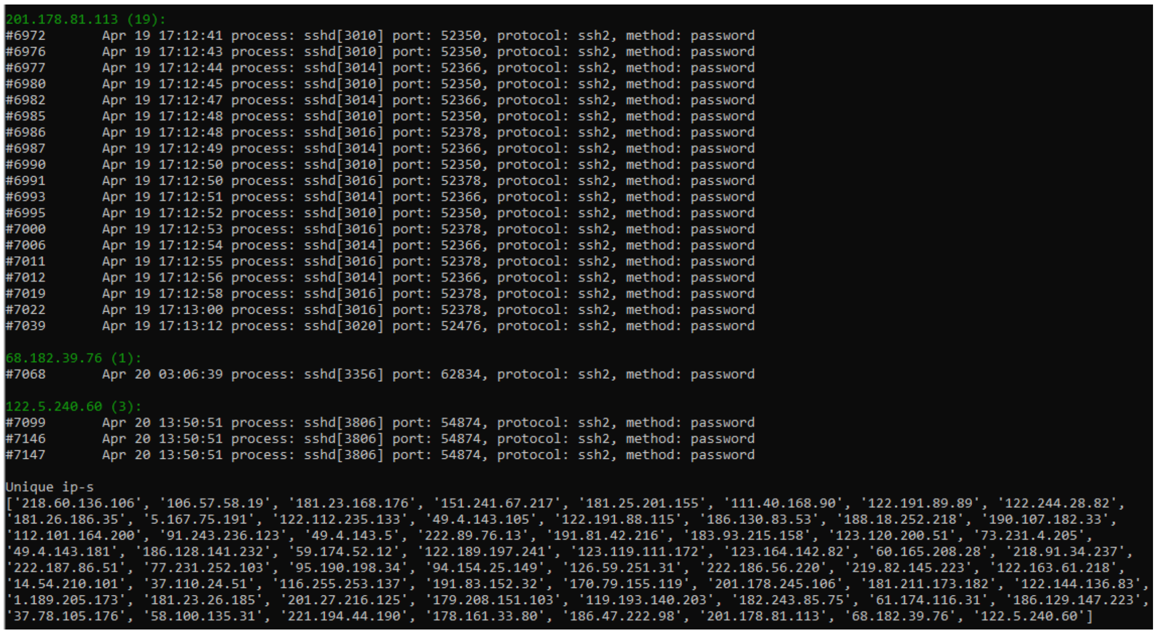
- **2) Show Command History for User:** This option is specifically designed to display the history of `sudo` commands that were executed by a given user. This is particularly useful for auditing administrative actions and identifying potentially risky command executions.
  Example: To view the `sudo` command history for the user "root," you would enter `2 root` and press Enter.



```
#1016     Mar 28 19:13:44   Directory: /home/ubuntu                                   /usr/sbin/service ssh restart
#1033     Mar 28 19:15:21   Directory: /home/ubuntu                                   /usr/sbin/service metricbeat stop
#1036     Mar 28 19:15:52   Directory: /home/ubuntu                                   /usr/bin/vim /etc/metricbeat/metricbeat.yml
#1039     Mar 28 19:16:14   Directory: /home/ubuntu                                   /usr/sbin/service metricbeat start
#1042     Mar 28 19:16:27   Directory: /home/ubuntu                                   /usr/bin/vim /etc/metricbeat/metricbeat.yml
#1496     Mar 29 10:36:43   Directory: /home/ubuntu/misc_scripts                      ./create_n_users.sh
#1529     Mar 29 10:36:52   Directory: /home/ubuntu/misc_scripts                      ./create_n_users.sh
#1552     Mar 29 10:37:34   Directory: /home/ubuntu/misc_scripts                      ./create_n_users.sh
#1575     Mar 29 10:38:05   Directory: /home/ubuntu/misc_scripts                      ./create_n_users.sh
#1600     Mar 29 10:42:02   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/ssh/sshd_config
#1603     Mar 29 10:42:24   Directory: /home/ubuntu/misc_scripts                      /usr/sbin/service sshd restart
#1695     Mar 29 11:39:18   Directory: /home/ubuntu                                   /bin/su
#1741     Mar 29 12:10:04   Directory: /home/ubuntu                                   /usr/sbin/groupadd elastic_users
#2565     Mar 30 13:06:28   Directory: /home/ubuntu                                   /bin/su
#3905     Mar 31 11:00:03   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/resolvconf/resolv.conf.d/base
#3908     Mar 31 11:00:22   Directory: /home/ubuntu/misc_scripts                      /sbin/resolvconf -u
#3957     Mar 31 11:34:22   Directory: /home/ubuntu/misc_scripts                      /usr/bin/hexdump
#4403     Apr  3 10:39:07   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/hosts
#4406     Apr  3 10:40:10   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/hosts
#4409     Apr  3 10:42:51   Directory: /home/ubuntu/misc_scripts                      /usr/bin/apt-get install Dnsmasq
#4412     Apr  3 10:42:54   Directory: /home/ubuntu/misc_scripts                      /usr/bin/apt-get install dnsmasq
#4415     Apr  3 10:49:39   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/resolv.conf
#4418     Apr  3 10:50:53   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/dnsmasq.conf
#4421     Apr  3 10:51:32   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/dnsmasq.conf
#4424     Apr  3 10:52:08   Directory: /home/ubuntu/misc_scripts                      /usr/sbin/service dnsmasq restart
#4427     Apr  3 11:00:16   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/resolv.conf
#4430     Apr  3 11:00:44   Directory: /home/ubuntu/misc_scripts                      /usr/bin/vim /etc/resolv.conf
```

- **3) Show Failed Login Attempts for User by IP:** This option lists all the failed login attempts for a specified user, and it organizes these attempts by the IP address from which they originated. This is crucial for security analysis, as it can help identify potential brute-force attacks or other unauthorized access attempts coming from specific sources.

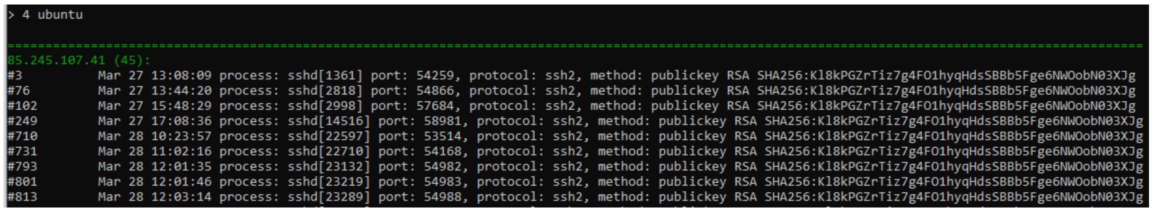
  Example: To see the failed login attempts for the user "testuser," you would enter `3 testuser` and press Enter.



```
201.178.81.113 (19):
#6972      Apr 19 17:12:41 process: sshd[3010] port: 52350, protocol: ssh2, method: password
#6976      Apr 19 17:12:43 process: sshd[3010] port: 52350, protocol: ssh2, method: password
#6977      Apr 19 17:12:44 process: sshd[3014] port: 52366, protocol: ssh2, method: password
#6980      Apr 19 17:12:45 process: sshd[3010] port: 52350, protocol: ssh2, method: password
#6982      Apr 19 17:12:47 process: sshd[3014] port: 52366, protocol: ssh2, method: password
#6985      Apr 19 17:12:48 process: sshd[3010] port: 52350, protocol: ssh2, method: password
#6986      Apr 19 17:12:48 process: sshd[3016] port: 52378, protocol: ssh2, method: password
#6987      Apr 19 17:12:49 process: sshd[3014] port: 52366, protocol: ssh2, method: password
#6990      Apr 19 17:12:50 process: sshd[3010] port: 52350, protocol: ssh2, method: password
#6991      Apr 19 17:12:50 process: sshd[3016] port: 52378, protocol: ssh2, method: password
#6993      Apr 19 17:12:51 process: sshd[3014] port: 52366, protocol: ssh2, method: password
#6995      Apr 19 17:12:52 process: sshd[3010] port: 52350, protocol: ssh2, method: password
#7000      Apr 19 17:12:53 process: sshd[3016] port: 52378, protocol: ssh2, method: password
#7006      Apr 19 17:12:54 process: sshd[3014] port: 52366, protocol: ssh2, method: password
#7011      Apr 19 17:12:55 process: sshd[3016] port: 52378, protocol: ssh2, method: password
#7012      Apr 19 17:12:56 process: sshd[3014] port: 52366, protocol: ssh2, method: password
#7019      Apr 19 17:12:58 process: sshd[3016] port: 52378, protocol: ssh2, method: password
#7022      Apr 19 17:13:00 process: sshd[3016] port: 52378, protocol: ssh2, method: password
#7039      Apr 19 17:13:12 process: sshd[3020] port: 52476, protocol: ssh2, method: password

68.182.39.76 (1):
#7068      Apr 20 03:06:39 process: sshd[3356] port: 62834, protocol: ssh2, method: password

122.5.240.60 (3):
#7099      Apr 20 13:50:51 process: sshd[3806] port: 54874, protocol: ssh2, method: password
#7146      Apr 20 13:50:51 process: sshd[3806] port: 54874, protocol: ssh2, method: password
#7147      Apr 20 13:50:51 process: sshd[3806] port: 54874, protocol: ssh2, method: password

Unique ip-s
['218.60.136.106', '106.57.58.19', '181.23.168.176', '151.241.67.217', '181.25.201.155', '111.40.168.90', '122.191.89.89', '122.244.28.82',
'181.26.186.35', '5.167.75.191', '122.112.235.133', '49.4.143.105', '122.191.88.115', '186.130.83.53', '188.18.252.218', '190.107.182.33',
'112.101.164.200', '91.243.236.123', '49.4.143.5', '222.89.76.13', '191.81.42.216', '183.93.215.158', '123.120.200.51', '73.231.4.205',
'49.4.143.181', '186.128.141.232', '59.174.52.12', '122.189.197.241', '123.119.111.172', '123.164.142.82', '60.165.208.28', '218.91.34.237',
'222.187.86.51', '77.231.252.103', '95.190.198.34', '94.154.25.149', '126.59.251.31', '222.186.56.220', '219.82.145.223', '122.163.61.218',
'14.54.210.101', '37.110.24.51', '116.255.253.137', '191.83.152.32', '170.79.155.119', '201.178.245.106', '181.211.173.182', '122.144.136.83',
'1.189.205.173', '181.23.26.185', '201.27.216.125', '179.208.151.103', '119.193.140.203', '182.243.85.75', '61.174.116.31', '186.129.147.223',
'37.78.105.176', '58.100.135.31', '221.194.44.190', '178.161.33.80', '186.47.222.98', '201.178.81.113', '68.182.39.76', '122.5.240.60']
```

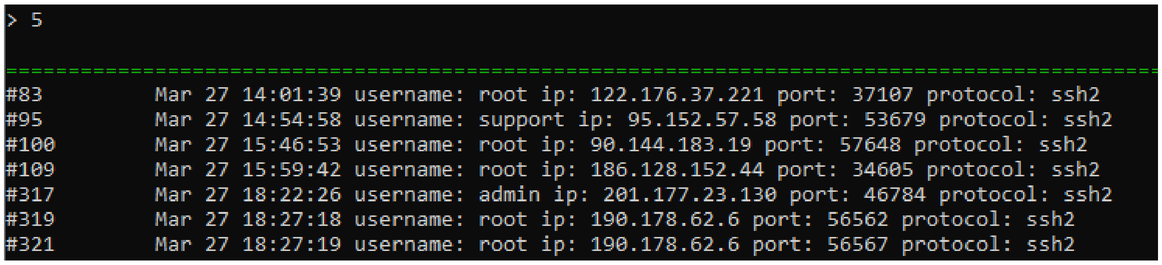
- **4) Show Successful Logins for User by IP:** Similar to the previous option, this one lists successful login attempts for a user, grouped by the source IP address. This helps in tracking user access patterns and identifying potentially suspicious logins from unfamiliar locations.

  Example: To see the successful logins for the user "ubuntu," you would enter `4` ubuntu and press Enter.



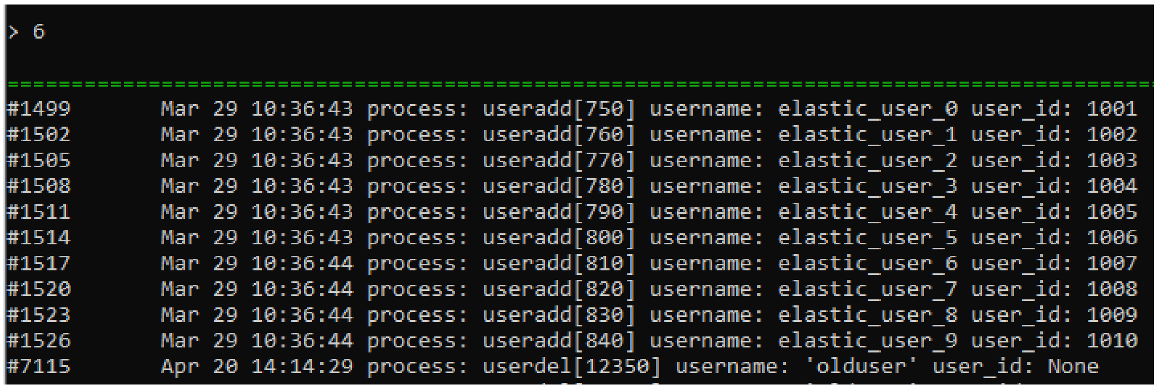
```
> 4 ubuntu

==========================================================================================================================
85.245.107.41 (45):
#3          Mar 27 13:08:09 process: sshd[1361] port: 54259, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#76         Mar 27 13:44:20 process: sshd[2818] port: 54866, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#102        Mar 27 15:48:29 process: sshd[2998] port: 57684, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#249        Mar 27 17:08:36 process: sshd[14516] port: 58981, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#710        Mar 28 10:23:57 process: sshd[22597] port: 53514, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#731        Mar 28 11:02:16 process: sshd[22710] port: 54168, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#793        Mar 28 12:01:35 process: sshd[23132] port: 54982, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#801        Mar 28 12:01:46 process: sshd[23219] port: 54983, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
#813        Mar 28 12:03:14 process: sshd[23289] port: 54988, protocol: ssh2, method: publickey RSA SHA256:Kl8kPGZrTiz7g4FO1hyqHdsSBBb5Fge6NWOobN03XJg
```

- **5) Show Max Authentication Attempts Exceeded Events:** This option displays log entries that indicate when a user exceeded the maximum number of allowed authentication attempts. These events are strong indicators of potential brute-force attacks or attempts to guess user passwords.



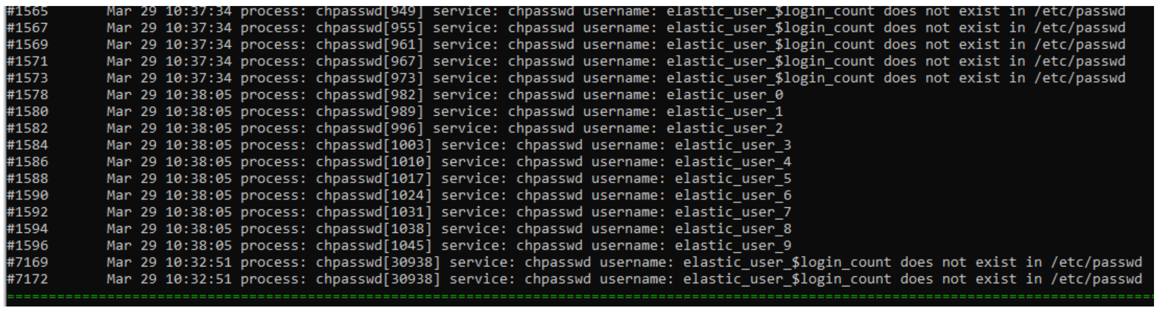
```
> 5

==========================================================================================
#83         Mar 27 14:01:39 username: root ip: 122.176.37.221 port: 37107 protocol: ssh2
#95         Mar 27 14:54:58 username: support ip: 95.152.57.58 port: 53679 protocol: ssh2
#100        Mar 27 15:46:53 username: root ip: 90.144.183.19 port: 57648 protocol: ssh2
#109        Mar 27 15:59:42 username: root ip: 186.128.152.44 port: 34605 protocol: ssh2
#317        Mar 27 18:22:26 username: admin ip: 201.177.23.130 port: 46784 protocol: ssh2
#319        Mar 27 18:27:18 username: root ip: 190.178.62.6 port: 56562 protocol: ssh2
#321        Mar 27 18:27:19 username: root ip: 190.178.62.6 port: 56567 protocol: ssh2
```

- **6) Show Create and Delete User Events:** This option provides a focused view of user account management activities, specifically showing when user accounts were created or deleted. This is useful for auditing account changes and ensuring proper user lifecycle management.



```
> 6

==========================================================================================
#1499       Mar 29 10:36:43 process: useradd[750] username: elastic_user_0 user_id: 1001
#1502       Mar 29 10:36:43 process: useradd[760] username: elastic_user_1 user_id: 1002
#1505       Mar 29 10:36:43 process: useradd[770] username: elastic_user_2 user_id: 1003
#1508       Mar 29 10:36:43 process: useradd[780] username: elastic_user_3 user_id: 1004
#1511       Mar 29 10:36:43 process: useradd[790] username: elastic_user_4 user_id: 1005
#1514       Mar 29 10:36:43 process: useradd[800] username: elastic_user_5 user_id: 1006
#1517       Mar 29 10:36:44 process: useradd[810] username: elastic_user_6 user_id: 1007
#1520       Mar 29 10:36:44 process: useradd[820] username: elastic_user_7 user_id: 1008
#1523       Mar 29 10:36:44 process: useradd[830] username: elastic_user_8 user_id: 1009
#1526       Mar 29 10:36:44 process: useradd[840] username: elastic_user_9 user_id: 1010
#7115       Apr 20 14:14:29 process: userdel[12350] username: 'olduser' user_id: None
```

- **7) Show Password Change Events:** This option displays log entries related to user password changes. This is important for security auditing and ensuring that password policies are being followed.



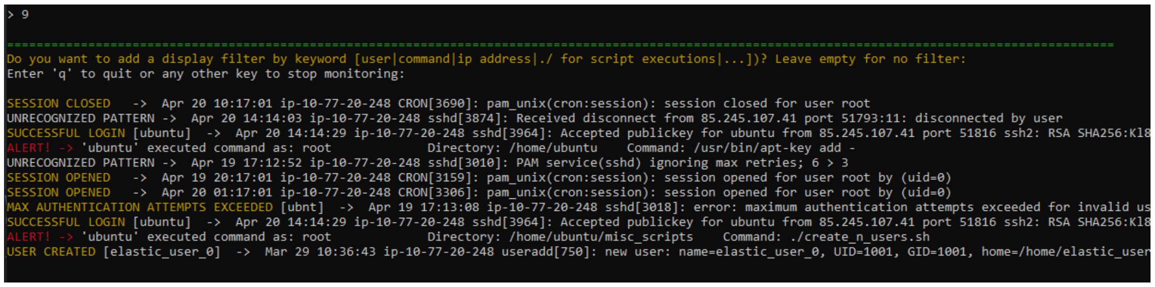
```
#1565      Mar 29 10:37:34 process: chpasswd[949] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
#1567      Mar 29 10:37:34 process: chpasswd[955] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
#1569      Mar 29 10:37:34 process: chpasswd[961] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
#1571      Mar 29 10:37:34 process: chpasswd[967] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
#1573      Mar 29 10:37:34 process: chpasswd[973] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
#1578      Mar 29 10:38:05 process: chpasswd[982] service: chpasswd username: elastic_user_0
#1580      Mar 29 10:38:05 process: chpasswd[989] service: chpasswd username: elastic_user_1
#1582      Mar 29 10:38:05 process: chpasswd[996] service: chpasswd username: elastic_user_2
#1584      Mar 29 10:38:05 process: chpasswd[1003] service: chpasswd username: elastic_user_3
#1586      Mar 29 10:38:05 process: chpasswd[1010] service: chpasswd username: elastic_user_4
#1588      Mar 29 10:38:05 process: chpasswd[1017] service: chpasswd username: elastic_user_5
#1590      Mar 29 10:38:05 process: chpasswd[1024] service: chpasswd username: elastic_user_6
#1592      Mar 29 10:38:05 process: chpasswd[1031] service: chpasswd username: elastic_user_7
#1594      Mar 29 10:38:05 process: chpasswd[1038] service: chpasswd username: elastic_user_8
#1596      Mar 29 10:38:05 process: chpasswd[1045] service: chpasswd username: elastic_user_9
#7169      Mar 29 10:32:51 process: chpasswd[30938] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
#7172      Mar 29 10:32:51 process: chpasswd[30938] service: chpasswd username: elastic_user_$login_count does not exist in /etc/passwd
==============================================================================================================================
```

- **8) Print Report Again:** This option allows you to reprint the initial analysis report. This can be useful if you need to refer back to the report after using other menu options.

- **9) Monitor the Log File in Real-Time:** This powerful option enables you to monitor the log file as it is being written to, providing live updates on authentication-related events.

  You also have the option to provide a filter keyword. If you enter a keyword (e.g., "user", "command", "ip address", or even a specific pattern like "./" to look for script executions), the monitoring will only display log entries that contain that keyword. This allows you to focus on specific types of events or activities. If you leave the filter empty, all new log entries will be displayed. If you don't want to filter just press ENTER.

  To stop the real-time monitoring, you need to type the letter 'q' and then press Enter.



```
> 9

=====================================================================================================================
Do you want to add a display filter by keyword [user|command|ip address|./ for script executions|...])? Leave empty for no filter:
Enter 'q' to quit or any other key to stop monitoring:

SESSION CLOSED   ->  Apr 20 10:17:01 ip-10-77-20-248 CRON[3690]: pam_unix(cron:session): session closed for user root
UNRECOGNIZED PATTERN ->  Apr 20 14:14:03 ip-10-77-20-248 sshd[3874]: Received disconnect from 85.245.107.41 port 51793:11: disconnected by user
SUCCESSFUL LOGIN [ubuntu]  ->  Apr 20 14:14:29 ip-10-77-20-248 sshd[3964]: Accepted publickey for ubuntu from 85.245.107.41 port 51816 ssh2: RSA SHA256:Kl8
ALERT! -> 'ubuntu' executed command as: root            Directory: /home/ubuntu    Command: /usr/bin/apt-key add -
UNRECOGNIZED PATTERN ->  Apr 19 17:12:52 ip-10-77-20-248 sshd[3010]: PAM service(sshd) ignoring max retries; 6 > 3
SESSION OPENED   ->  Apr 19 20:17:01 ip-10-77-20-248 CRON[3159]: pam_unix(cron:session): session opened for user root by (uid=0)
SESSION OPENED   ->  Apr 20 01:17:01 ip-10-77-20-248 CRON[3306]: pam_unix(cron:session): session opened for user root by (uid=0)
MAX AUTHENTICATION ATTEMPTS EXCEEDED [ubnt]  ->  Apr 19 17:13:08 ip-10-77-20-248 sshd[3018]: error: maximum authentication attempts exceeded for invalid us
SUCCESSFUL LOGIN [ubuntu]  ->  Apr 20 14:14:29 ip-10-77-20-248 sshd[3964]: Accepted publickey for ubuntu from 85.245.107.41 port 51816 ssh2: RSA SHA256:Kl8
ALERT! -> 'ubuntu' executed command as: root            Directory: /home/ubuntu/misc_scripts    Command: ./create_n_users.sh
USER CREATED [elastic_user_0]  ->  Mar 29 10:36:43 ip-10-77-20-248 useradd[750]: new user: name=elastic_user_0, UID=1001, GID=1001, home=/home/elastic_user
```

- **q) Quit:** This option allows you to exit the program and return to the command prompt or terminal.

Example Usage Scenarios

Here are some more detailed examples of how to use the script:

Showing User Command History

To view a detailed command history for the user "john":

1. Run the script by typing `python3 auth_log_analyzer.py` in your terminal and pressing Enter.
2. The script will prompt you for the log file path. Provide the correct path and press Enter.
3. The script will process the log file.
4. Once the menu appears, enter `2 john` and press Enter.
5. The script will then display a list of all the `sudo` commands that were executed by the user "john." Each command will be shown with its associated timestamp and the directory from which the command was executed. This allows you to see not only *what* commands were run, but also *when* and *from where*.

Showing Failed Login Attempts

To view a comprehensive list of failed login attempts for the user "testuser":

1. Execute the script using `python3 auth_log_analyzer.py` in your terminal and press Enter.
2. Enter the path to the `auth.log` file when prompted and press Enter.
3. The script will process the log file.
4. When the menu is displayed, type `3 testuser` and press Enter.
5. The script will present a detailed list of failed login attempts for the user "testuser." This list will be organized by the IP address from which the failed attempts originated. For each failed attempt, the script will provide information such as the timestamp of the attempt, the process that was involved (e.g., sshd), the port used for the connection, the authentication method that was used (e.g., password, publickey), and any SSH signature information if available. This level of detail is critical for understanding the nature and source of potential attacks.

Showing Create and Delete User Events

To display a detailed log of user creation and deletion events:

1. Run the script by typing `python3 auth_log_analyzer.py` and pressing Enter in your terminal.
2. Provide the path to the `auth.log` file when prompted and press Enter.
3. The script will process the log file.
4. When the menu is displayed, enter `6` and press Enter.
5. The script will display a detailed list of user creation and deletion events. For each event, the script will provide information such as the timestamp of the event, the process that was responsible for the event, the username of the user that was created or deleted, and the user ID (UID) associated with the user. This allows for a clear audit trail of user account changes.

Live Monitoring

To monitor the log file in real-time for new authentication events:

1. Start the script by typing `python3 auth_log_analyzer.py` in your terminal and pressing Enter.
2. Enter the log file path when prompted and press Enter.
3. The script will process the log file.
4. When the menu appears, type `9` and press Enter.
5. The script will then prompt you if you want to add a filter. If you want to monitor all activity, just press Enter. If you want to filter for specific activity, enter a keyword like "sudo", "user", an IP address, or a command and then press Enter. Keep in mind that the filter keyword is case-sensitive.
6. The script will now begin monitoring the log file in real-time. Any new lines that are written to the log file will be processed by the script. If a filter keyword was provided, only the lines that contain that keyword will be displayed.
7. The script will display different types of events with color-coding to help you quickly identify them. For example, `sudo` commands may be displayed in red as alerts.
8. To stop the monitoring process, type the letter `q` and press Enter. The script will then stop monitoring the log file and return you to the menu.

Important Notes

- **Log Format Assumptions:** The script is designed to work with the standard format of `auth.log` files as they are typically generated by Linux systems. However, it's important to be aware that there might be slight variations in the log format depending on the specific system configuration or any customizations that might have been made. If the log format deviates significantly from the standard format, the script might not be able to parse the log data correctly, which could lead to inaccurate results. The tool will notify you if this is the case.
- **Exclusion of Insignificant Entries:** To enhance the clarity and focus of the analysis, the script is designed to exclude certain log entries that are considered to be insignificant, duplicates to other entries or irrelevant to security analysis. These entries might include routine system messages or other informational logs that do not provide meaningful security-related insights. By excluding these entries, the script helps to reduce noise and highlight the more important events.
- **Color-Coded Output:** To improve the readability and visual clarity of the output in the terminal, the script utilizes color-coded text. Different types of information or events might be displayed in different colors, making it easier for the user to quickly scan the output and identify key information. This color-coding can be particularly helpful when monitoring the log file in real-time, as it can help to draw attention to important events.
- **The tool is a work in progress:** There is a lot of room for improvement and adding additional functionalities. Error handling needs to be improved in general. If you have any ideas or recommendations, feel free to ping me at radtonev@gmail.com or open a Github issue.