# Solving Kioptrix 4

This challenge is about using a vulnerable tool installed on the VM to edit files. It runs as root, and we can use this to modify the sudo command system. **Typing is critical, don’t make any typos.**

## Before you start

As always, consider a fresh instance of Kali.

Windows

VirtualBox

Host Only Network

Kioptrix 3

Kali

Internet

**NAT**

## VM Setup:

Once you have downloaded the VM from the link in Learn, deploy to your VirtualBox environment. Make sure there are not warnings about your virtual machine’s settings. The config you did for Kioptrix 1 should be good for this VM as well, and thanx to Mat in Section 2, you can just as easily swap out the .vmdk file from Kioptrix 1 and use the .vmdk from this challenge.

Unlike Kioptrix 1 and 2, we don’t need to use an IDE controller, a SATA controller works fine. Actually either a traditional IDE or a more modern SATA controller works for the hard drive. The other settings, however, should be verified.

## Scan:

Once the VM is up and running, scan the network, and detail scan the VM with the following:

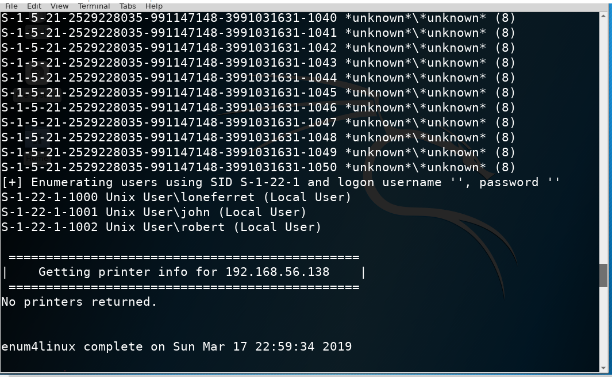
nmap –sV –O –P1-65534 <ip\_address>  
nmap –A <ip\_address> (we note that above, the default port range is sufficient)

## Samba Scan

Scan the samba service with enum4linux:

enum4linux <ip\_address>

As we have seen in the past, you may discover a legacy version of samba running. You might need to modify your /etc/samba/smb.conf file to allow legacy support of NT1 servers. Once configured, you should see at the bottom of the result a list of users that might be of value:

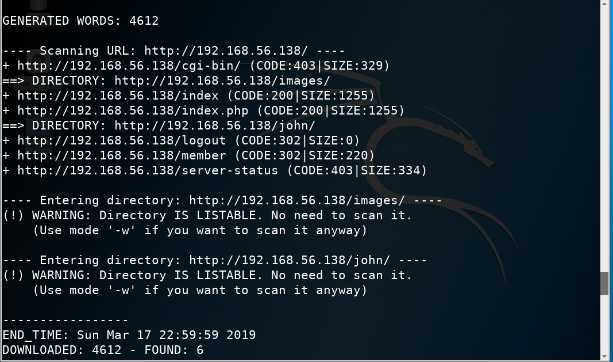


## Website Scan

Now we know there is a website, and we have configured Kali to use a domain name, and not an IP address, we can scan available website content.

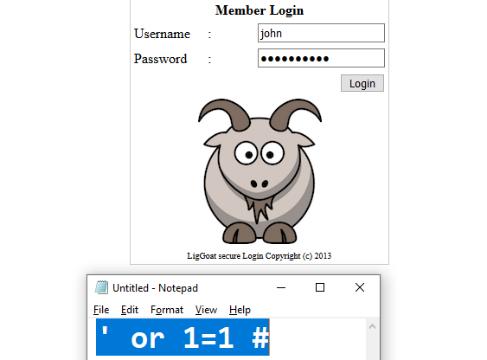
dirb <ip\_address>  
nikto <ip\_address>

dirb returns a lot of good result, and nikto returns a list of standard items. We check out the directories it mentions



Looks like we have a website that can be exploited, and usernames we can try.

Let’s try the login page with standard MySQL based SQL injections **' or 1=1 #:**



Given the password is hashed out like a proper password field on a webform, it will be easier to use notepad, as we have above. Just copy/paste the data to the password field.

We see that John’s password is stored in plain text. Let’s try with the rest of the users and see what we get.

Now we could try and cracking the passwords, however, we see we have a plain text password for John. Let’s try and use this to log in:

root@kali:~# ssh john@<ip\_address>

john@192.168.182.154's password:

Welcome to LigGoat Security Systems - We are Watching

== Welcome LigGoat Employee ==

LigGoat Shell is in place so you don't screw up

Type '?' or 'help' to get the list of allowed commands

john:~$ ?

cd clear echo exit help ll lpath ls

john:~$

This shell doesn’t look very good, it is purposely limited to prevent “screw ups”, but it does allow echo. We saw when we did remote shells earlier the ability to run your own shell, and one of the commands was: **echo os.system('/bin/bash')**

Now we have an easier shell to use.

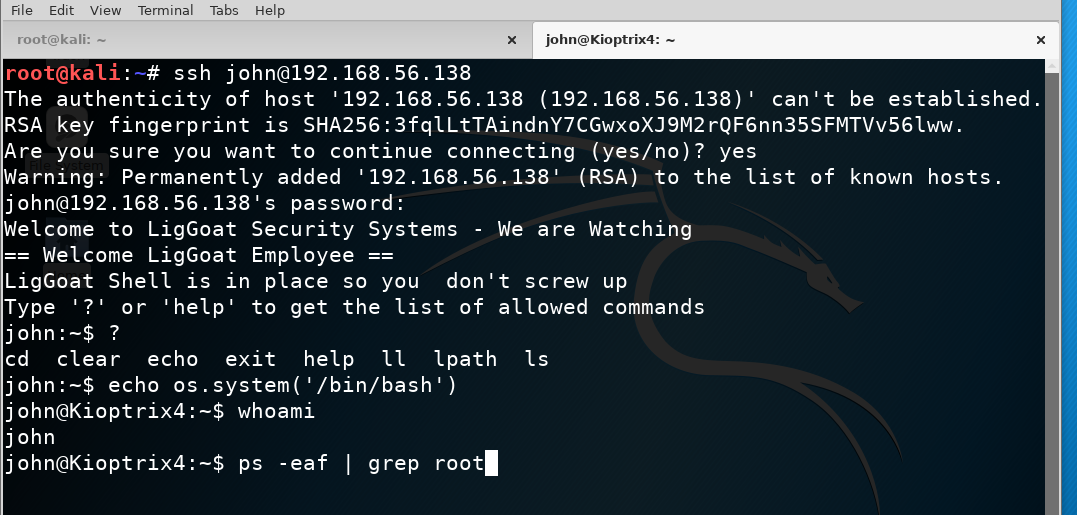
First, let’s see if sudo is available.

sudo -v

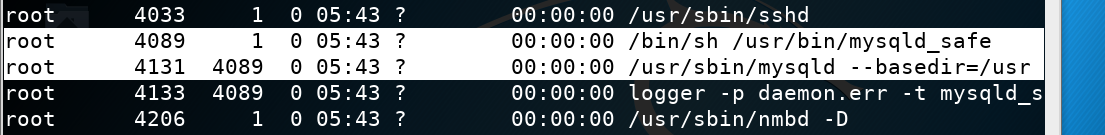
This will test to see if sudo is installed. It appears to be, so we don’t need to try and supply a password. Hit **Ctrl + C** to exit if necessary.

There are several things we can try, but we are going to look for processes running as root. Type in the following:

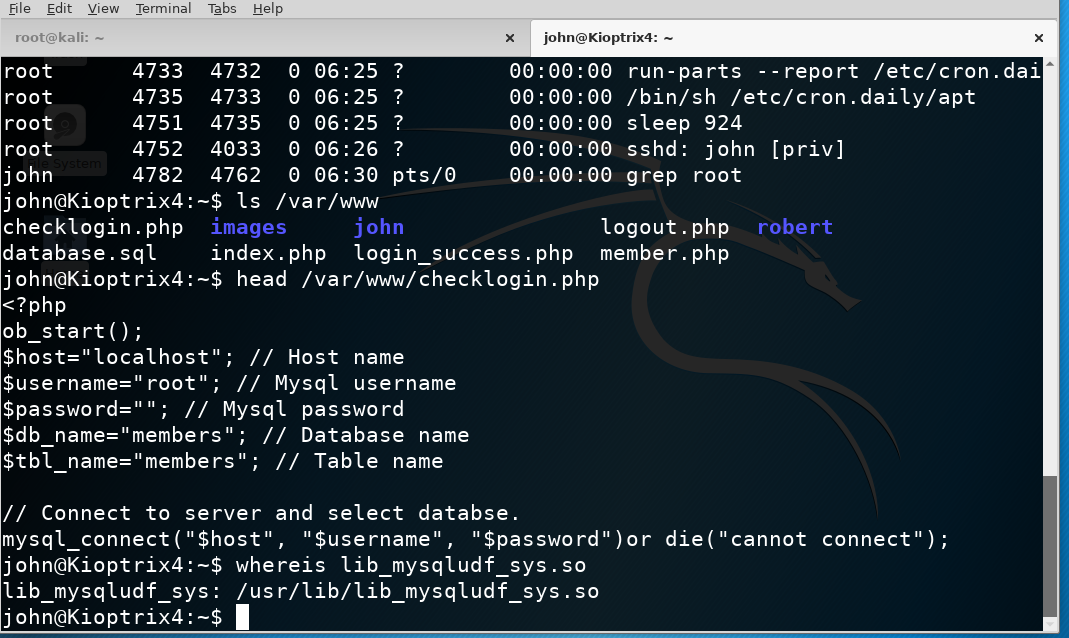
ps –eaf | grep root



You should see a list of processes run by root. One of which is MySQL, which as we have discussed before, is very wrong!!!!!



If we can log into MySQL, and escape out to the shell. If we can escape out to the shell, that shell becomes root. First find the connection settings of the web application (found in /var/www) and we see a checklogin page. If we look at this file (you can use cat, I just grabbed the first 10 lines for demonstration purposes) you see the connection info. Finally below, we are verifying the mysqludf module is installed. We do this with the whereis command below:



Now let’s try a MySQL session and exploit the fact it is running as root. In the same tab (same remote session of course) log into MySQL as below (no password required), execute the command to add John (you are currently logged in as John) to the admin group, and verify who you are at this time.

The command you are executing is the usermod command and it adds you to the admin group. The admin group can then run sudo, and as we have seen, we can run sudo su to become root.

Again, this is why it is of critical import you do not run server processes like MySQL as root, as it allows someone with the credentials to run commands as root. Normally, you wouldn’t be able to just add yourself to an admin group, but because we ran it as root, we could. You will need to enter John’s password again, as you discovered above.

See below for the final steps:

