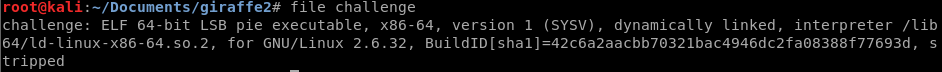
­ Another day, another challenge from our favorite hacker giraffe. Here is the challenge prompt this time:





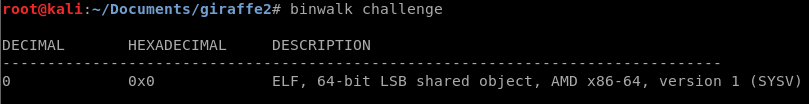
Downloading from the given the link, we get the file **challenge**. Since it has no extension, we can check the filetype with **file**. L



Looks like an executable, let’s use **chmod** to allow execution and **./challenge** run it.



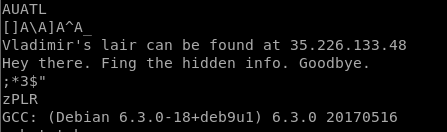
No hints here, except that some info is hidden. Just in case, let’s use **binwalk** to check for hidden files (not likely, binwalk generally only works on images)



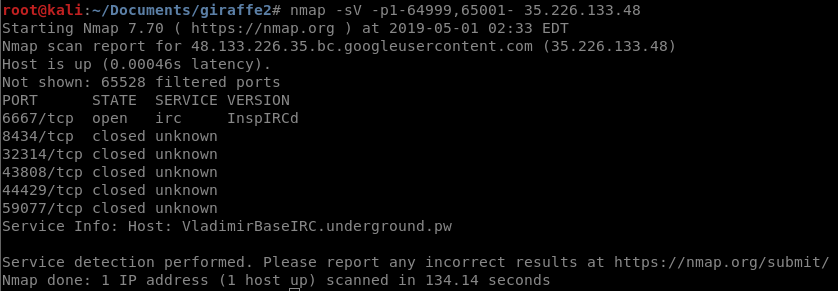
Sure enough, it is just an elf executable. The next step is to check the binary for readable ascii text with the **strings** command.



We get a whole bunch, so I will skip to the obviously important part:

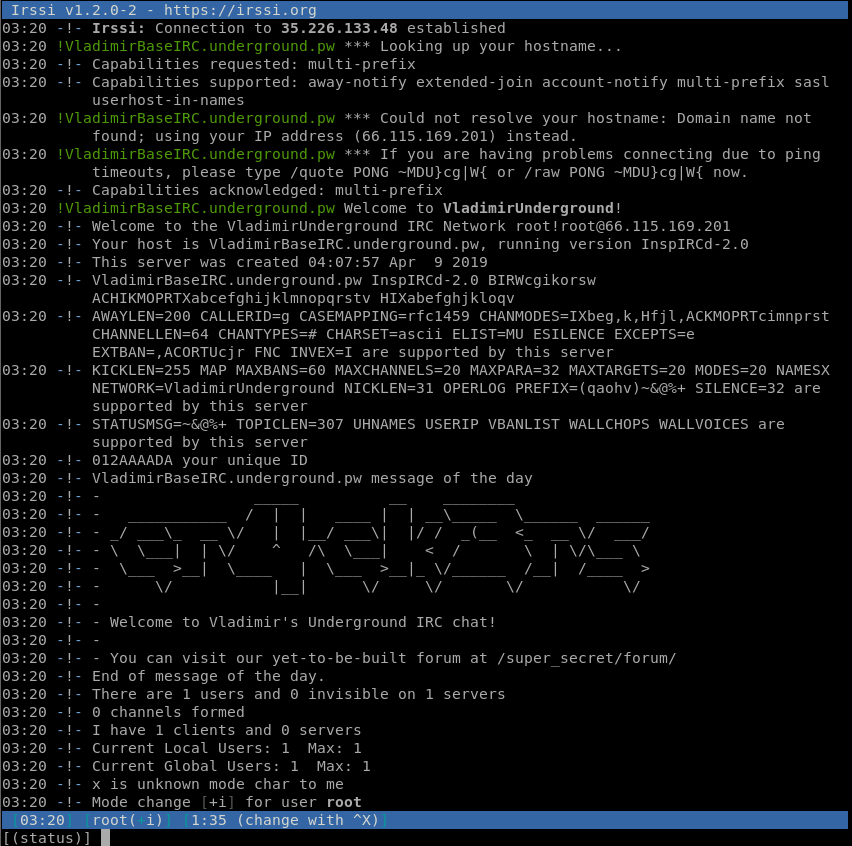


Great, we have an IP address. Time to scan it!



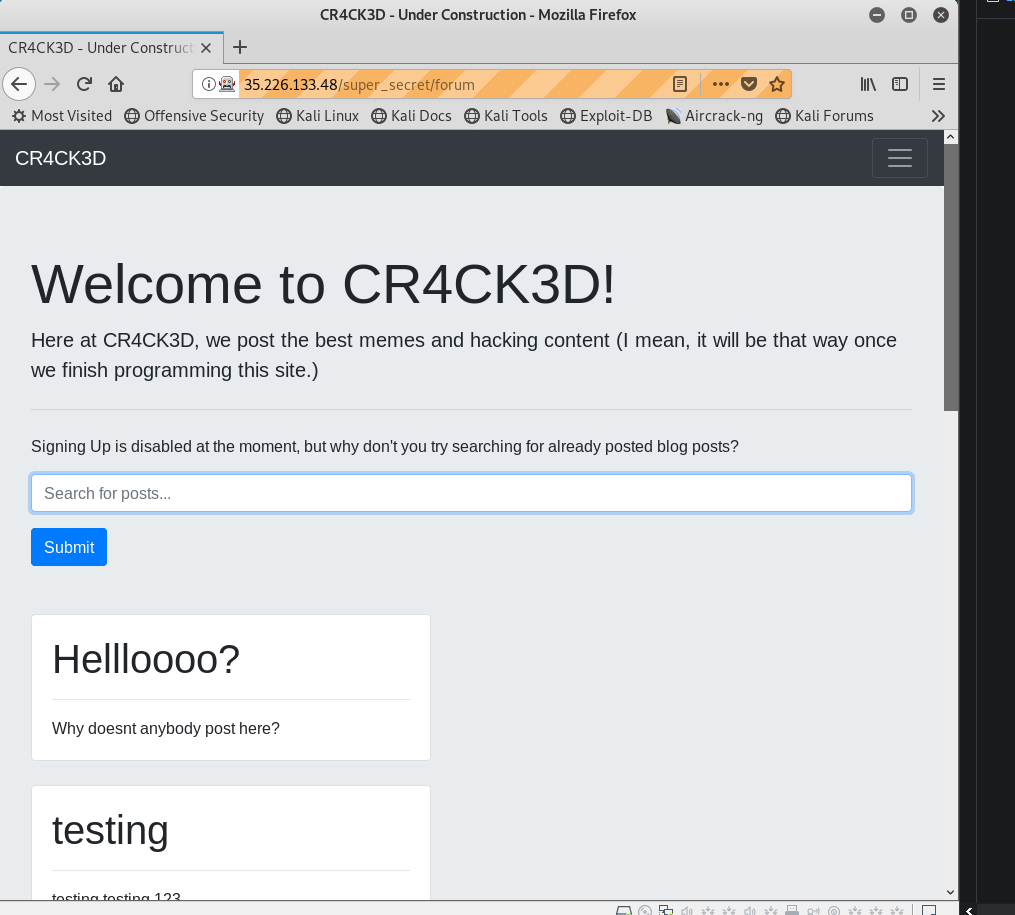
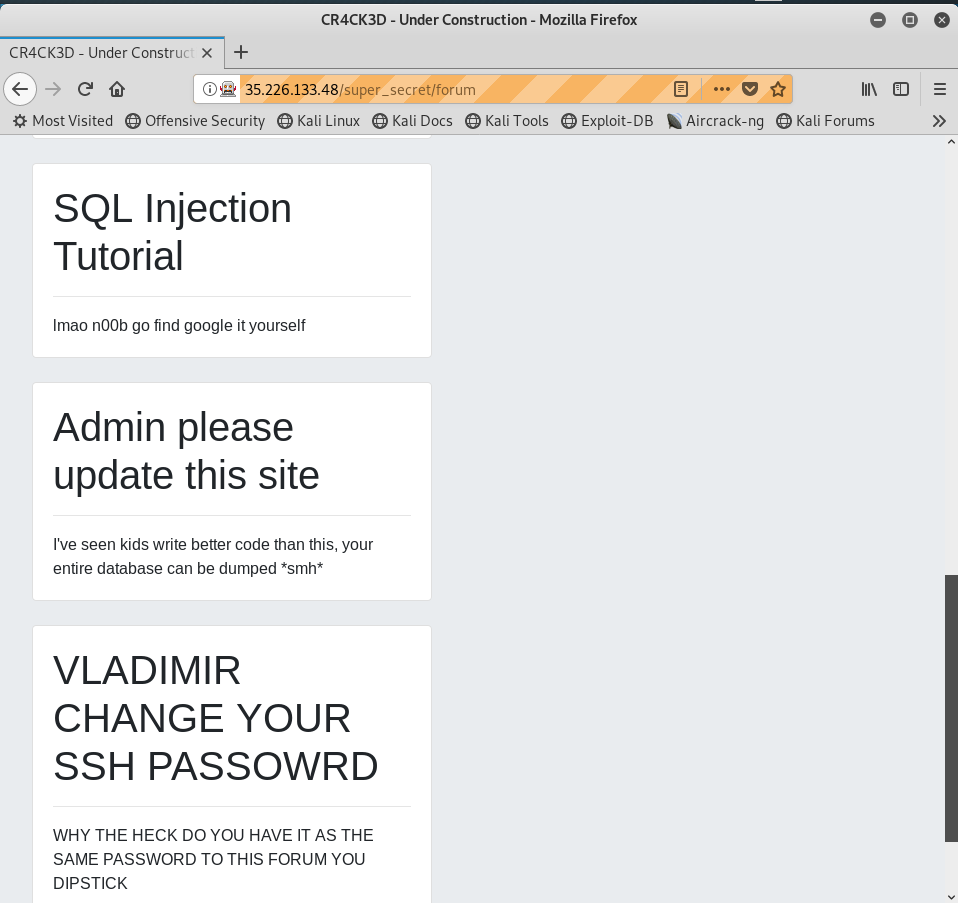
So we find some interesting info, including an **IRC** (internet relay chat) service running on port 6667. There is also a service info message giving us a message that the host is VladmirBaseIRC.ungerground.pw

We can connect to the irc service using the command **irssi**. You can read the full usage with man **irssi**, but the main syntax is **irssi -c <ip address> -p <port> -h <host>,** which in this case is **irssi 35.226.133.48 -p 6667 -h VladmirBaseIRC.ungerground.pw**



So now we are in the IRC server. I don’t actually know how to use IRC at this point, so I look it up real fast. You can see from the image above giving us a hint to look in /super\_secret/forum/.

On OWASP Zap, I navigate to the directory given in the message using a proxied browser.

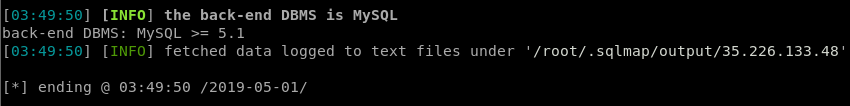
 

A rather basic site, but perhaps it is vulnerable to SQL injection. It also gives us valuable information about the site’s setup, such as the fact that Vladmir likes to use default passwords (despite claiming to be the best hacker ever). I will start a **dirb** scan on the service, as well as an **sqlmap** scan as the service is likely to be vulnerable.

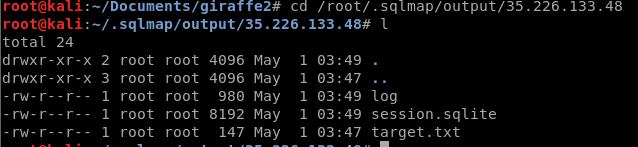
The way this works is, since the website has a function to query the database (the bar at the top), we can potentially exploit vulnerabilities in the backend to get the server to spit out all its database data. In order to get the form parameters, I entered a search for “anything” and copied the resulting url into sqlmap.



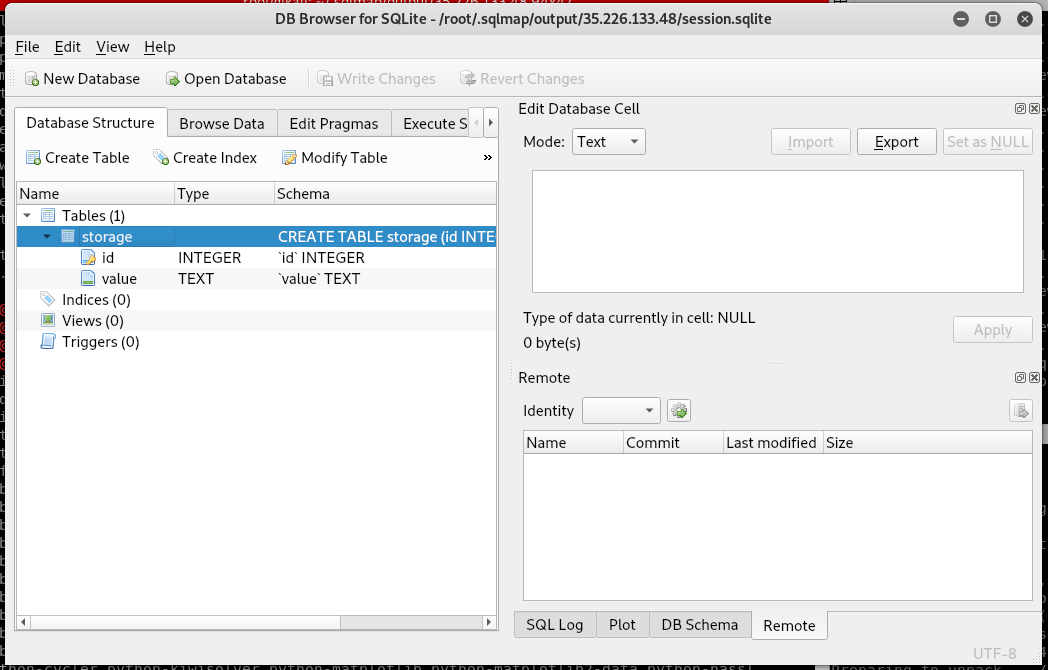
The scan then begins doing stuff. I generally hit “yes” for any prompts that show up. Once the scan finishes, the retrieved data is logged to a directory in your user’s home:



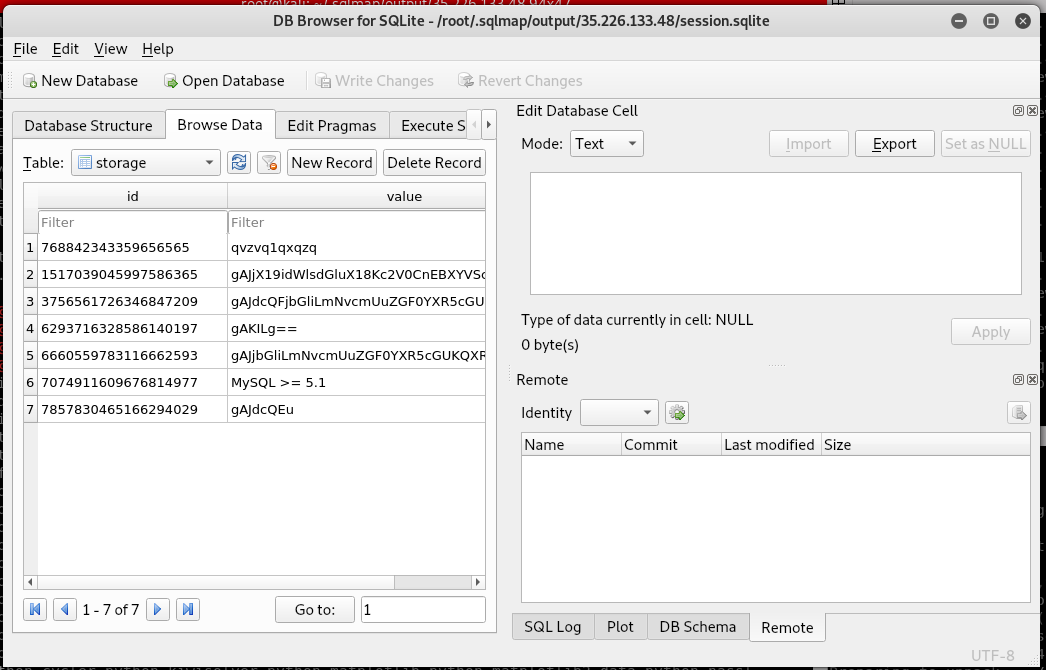
Let’s check what files we ~~stole~~ extracted:



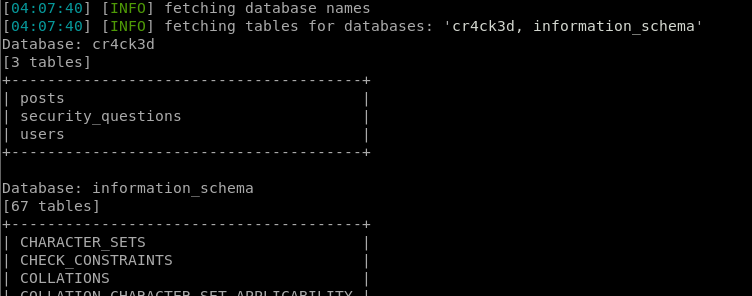
Looks like we got a **sqlite** session backup. If you don’t have sqlite, use **apt install sqlite** to install it real quick. You can open the file in the command line, but I used **sqlitebrowser** which has a nice gui.



As you can see from the **Database Structure** tab, the database has one table with two columns: id and value. Pretty basic. Let’s see what is in it using the **Browse Data** tab:

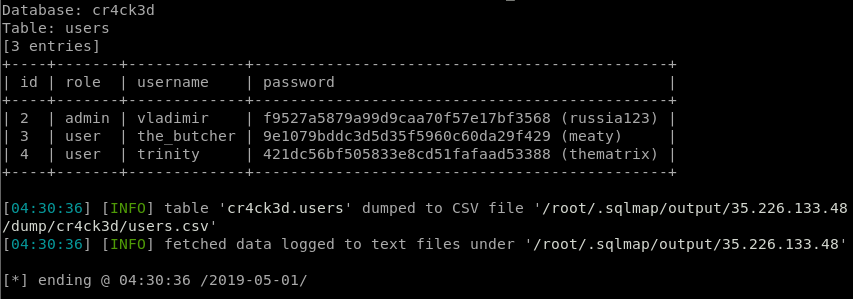


There are several base64 looking strings, mapped to random looking numbers. Let’s try and decode each of the lines to see what we can get… Well I tried it and it didn’t work. Time to backtrack: I re-ran **sqlmap**, this time with the **–tables** tag to resolve more tables. Here is a snippet of the results:

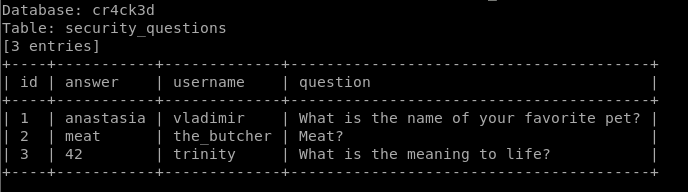


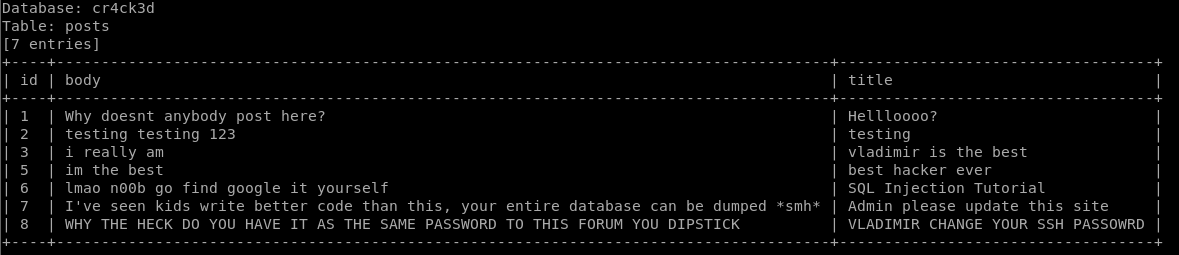
Hmmmm, a table called **users**? That looks interesting. It also includes security questions, which could be helpful later. Lets open these tables and see what we can get! We’ll need to run **sqlmap** again, this time with the **–columns -D cr4ck3d -T users** tags added to retrieve the database tables. I used about 10 threads when it asked, seemed like a reasonable number. This ended up making my requests time out fairly frequently but you can adjust the thread count for your own needs.

Once you have an idea of the table layout, it is time to get some data. Use the **–dump** tag for this.



YAY WE HAVE SWEET CREDENTIALS! Just for fun, lets dump the other tables:





We get some cute security questions and all of the posts from the main page. So where do we go next? Well, according to the forum post, Vlad’s **SSH** password is the same as his password for this database, so let’s go ahead and try logging in with it:



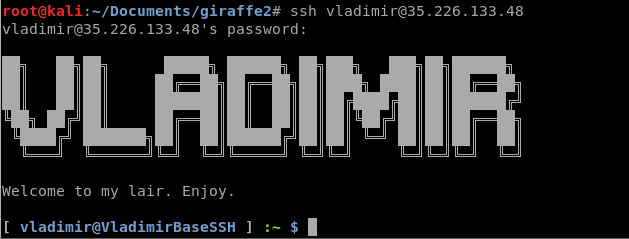
Well that didn’t work… lets try the other users, and if they don’t work then we probably need to try a different port.



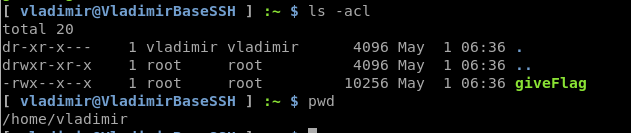


So we at least know that we got Vladimir’s password correct, since the other two users’ incorrect passwords give the expected error message.

Okay so apparently I just got a weird error, because I tried connecting again and got in just fine.



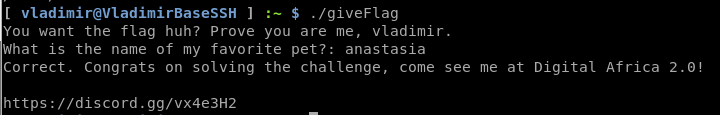
Now we are in an SSH server, lets run some basic enumeration



We are in Vlad’s home directory, and there is an executable file called giveFlag. Luckily, we have all the permissions we need to execute it, so lets do that.



Hey, we know what his favorite pet is from those security questions we dumped earlier! Let’s enter **Anastasia**!



And we did it! The discord is the same one we got into last time, but it’s not about the reward, its about the friends we made along the way! 😊