We Test Pens Incorporated

COMP90074 - Web Security Assignment 3

Xiande Wen

905003

**THREAT MODELLING REPORT FOR**

**Bank of UniMelb Pty. Ltd. - WEB APPLICATION**

**Report delivered: 03/06/2021**

# S: Spoofing

## Threat 1: Attacker authenticates as user after brute-forcing password

### Threat identified

|  |
| --- |
| An attacker could brute force a legitimate user’s password and authenticate into the application as the targeted user. This would result in a loss of personally identifiable information for the victim, and the attacker having the ability to perform actions (i.e., transferring money, closing the account, etc.) on the user’s behalf. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who tries to steal user’s money and personal information. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Require users to set complex passwords of at least 12 characters long with at least three-character sets; 2. Enable multi-factor authentication when users logging into the system; 3. Introduce rate limiting on the login page for usernames (account lockout after three unsuccessful attempts) and also for IP addresses to prevent an attacker targeting multiple usernames. |

## Threat 2: A logged in user changes another user’s password and authenticates into the system as that user

### Threat identified

|  |
| --- |
| Assuming the system has weak authentication in the password change functionality, a logged-in user could change another user’s password without authorisations and authenticate the system as the targeted user. This would result in a loss of personally identifiable information for the victim, and the attacker having the ability to perform actions (i.e., transferring money, closing the account, etc.) on the user’s behalf. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker with a set of valid credentials or a malicious user who tries to steal user’s money and personal information. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Ensure a strong authentication mechanism in the password change functionality: the user should only be able to change his/her own password with the correct old password provided. 2. Enable multi-factor authentications when changing the password and logging in. |

# T: Tampering

## Threat 1: An attacker logs into the system as a developer and alters the system code

### Threat identified

|  |
| --- |
| Assuming the system has a weak authentication in the developer login functionality, an attacker could authenticate into the system as a developer and alter the system code in malicious ways. This could result in the full system takeover, users’ financial losses and personal information breach. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker with a set of valid credentials or a malicious user who tries to take over the system and steal user’s money and personal information. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Remove the developer login function from the normal users’ view; 2. Enable multi-factor authentication and ensure a strong password complexity for the developer login function. |

## Threat 2: A malicious user can tamper with the cookies

### Threat identified

|  |
| --- |
| A malicious logged-in user could tamper with the Boolean value stored in the cookies which is used to identify if the current user is an admin user. This could result in the reveal of the admin panel, which could contain confidential information. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker with a set of valid credentials or a malicious user who tries to exploit the bank system for their own benefit (i.e., steal others’ money and personal information). |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Remove the cookie which is used to identify if the user is an admin from the client-side and ties it to the user’s session: check that by the user’s session id on the server-side; 2. Remove the admin panel entry from normal users’ view. 3. Restrict the normal users’ permission: normal users should not be allowed to promote users. |

# R: Repudiation

## Threat 1: An attacker repudiates that he/she did not log into others’ accounts to perform unauthorised actions

### Threat identified

|  |
| --- |
| Assuming the system does not properly log the password changing history (i.e., the user who performed this action is not logged), an attacker could repudiate that he/she took over others’ accounts and performed unauthorised actions, which he/she actually did. This could make it hard for the branch managers and admins to protect users’ interests and potentially involve the company in a law dispute. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who manages to take over others’ accounts and then steal others’ money and personal information. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Properly log the history for every performed action in the system; 2. Enable multi-factor authentication for password changing and logging in to make it hard to take over others’ accounts. |

## Threat 2: An attacker claims that he/she is the owner of an account

### Threat identified

|  |
| --- |
| Assume the system only has the username as the identifier for each user and does not properly log the password changing history, an attacker could claim that he/she is the owner of an account by providing both the username and the changed password. This could result in that the admin team has no ways to repudiate the lie and protect the victim user and potentially involve the company in a law dispute. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who manages to take over others’ accounts. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Properly log the history for every performed action; 2. Enable multi-factor authentication for password changing and logging in to make it hard to take over others’ accounts. |

# I: Information Disclosure

## Threat 1: An attacker acquires sensitive information from the files left during testing / development

### Threat identified

|  |
| --- |
| An attacker could view the files left during testing / development through the directory listing which might contain sensitive information. This could result in the system takeover and personal data breach. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who tries to exploit the system for their own benefit. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Remove those files from public view; 2. Disable that directory listing if it is not necessary. |

## Threat 2: An attacker acquires sensitive information from the admin panel

### Threat identified

|  |
| --- |
| Assume the admin panel contains some confidential information that is meant to be only visible to admins, the attacker could acquire that information after managed it to view the admin panel. This could result in the system takeover and personal data breach. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who tries to exploit the system for their own benefit. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Remove the cookie which is used to identify if the user is an admin from the client-side and ties it with user’s session: check that by the user’s session id on the server-side; 2. Remove the admin panel entry from normal users’ view. 3. Restrict the normal users’ permission: normal users should not be allowed to promote users. |

# D: Denial of Service

## Threat 1: An attacker takes down the server by sending large number of requests to the website concurrently

### Threat identified

|  |
| --- |
| Assuming the system has no rate limit on the number of requests from one IP address within a short period, an attacker could send a large number of requests concurrently to take down the server, which could result in denial of service. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who got paid from one of the company’s competitors to take down the server. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Introduce request rate limiting for IP addresses to prevent large number of requests sent from one place hence increase the cost of denial of service; 2. Increase the bandwidth to increase the cost of denial of service. |

## Threat 2: An attacker takes down the server by routing the traffic to other places

### Threat identified

|  |
| --- |
| Assuming an attacker authenticates as a developer and alters the code in a way that all traffic is redirected to other places, the system will no longer function correctly hence incur denial of service. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker who got paid from one of the company’s competitors to take down the server. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Remove the developer login function from the normal users’ view; 2. Enable multi-factor authentication and ensure a strong password complexity for the developer login function. |

# E: Elevation of Privilege

## Threat 1: An attacker authenticates into the system as a branch manager

### Threat identified

|  |
| --- |
| Assuming the system has weak authentication in the password change functionality, a logged-in user could change the branch manger’s password without authorisations and authenticate into the system as the branch manager. This could result in users’ financial losses, personal information breach and potentially system takeover. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker with a set of valid credentials or a malicious user who tries to steal user’s money and personal information. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Ensure a strong authentication mechanism in the password change functionality: the user should only be able to change his/her own password with the correct old password provided. 2. Enable multi-factor authentication when users trying to change their passwords or log in. |

## Threat 2: An attacker promotes him/herself to the system admin

### Threat identified

|  |
| --- |
| Assume the system has weak authentication in the user promotion functionality, a logged-in user could promote him/herself as the system admin without authorisations. This could result in users’ financial losses, personal information breach and system takeover. |

### Threat actor

|  |
| --- |
| The threat actor could be an attacker with a set of valid credentials or a malicious user who tries to take over the system and steal user’s money and personal information. |

### Threat remediation

|  |
| --- |
| Remediations for this threat would be to:   1. Remove the cookie which is used to identify if the user is an admin from the client-side and ties it with user’s session: check that by the user’s session id on the server-side; 2. Remove the admin panel entry from normal users’ view. 3. Restrict the normal users’ permission: normal users should not be allowed to promote users. |