SEC Consult
ADVISOR FOR YOUR INFORMATION SECURITY

Report 1804372

Web Application MBB eCustody

for

Malayan Banking Berhad

conducted by

SEC Consult

Version: 1.0
Responsible: W. Ikram
Confidentiality class: Strictly confidential

Date: 2019-10-31
Author: A. Zulkifli

Contents

1 Executive Summary

SEC Consult was appointed by Malayan Banking Berhad (Maybank) to perform a security audit on Web Application for MBB Trade Connex Malaysia application.

The following chapter summarizes the scope of the audit, the results of the audit and outlines the measures recommended by SEC Consult. Below table shows the vulnerability summary of the audited system for Maybank.

Risk assessment	No. of vulnerability classes
Low	1
Medium	2
High	0
Critical	0
Total	**3**

The graph below visualizes data from the table above.

1.1 Rules of Engagement

While conducting the security assessment, all SEC Consult security consultants shall adhere to the following rules:

1. Do not conduct Social Engineering attacks.
2. Do not conduct DoS or DDoS attacks.
3. Do not conduct tests/scans on servers/hosts that are out of scope.
4. Do not pass any information about the security assessment to third parties through any medium (unless officially requested by the client).
5. Do not perform any exploitation on the servers/hosts without clients' consent or agreement.

1.2 Scope of Work

The security assessment took place on 2019-10-30 and 2019-10-31. The objective of this test was to check the MBB eCustody Web application against all kinds of vulnerabilities and common configuration issues.

1.3 Audience

The intention of this document is to provide Maybank Management the information on possible risks and vulnerabilities inherent in the current web application. The content is proprietary and intended for internal use only. It should not be distributed or modified without the consent of Maybank and SEC Consult.

Please refer to Findings for more information on the observation from the security assessment.

2 Testing Methodology

SEC Consult conducts penetration tests to check the security of a complete system or single system components. The tools, methods and techniques used by SEC Consult fall into three categories:

1. Well known throughout both the computer security and "hacker" communities.[1]
2. In-house tools developed to extend the boundaries beyond the usual hacker's tool-kit.
3. Specialist observation. Examining the site to look for vulnerabilities that may not be discovered directly using tools.

3 Testing Guide

The system as defined in the permission to attack, has been tested against the following test classes:

3.1 Server Configuration

	Server configuration / generic	
	This class covers exploitable configuration errors for all kinds of server software.	
Attack pattern	Tested	Exploitable
Enumerating server contents	YES	NO
Exploiting default accounts	YES	NO
Enumerating user accounts	YES	NO
Exploiting dangerous protocol methods	YES	NO
Exploiting inappropriate access permissions	YES	NO
Exploiting unprotected functionality	YES	NO
Gathering internal information	YES	NO
Guessing passwords	YES	NO
Reading unencrypted sensitive data	YES	NO

3.2 Patch Level

	Server patch level	
	It is possible to exploit known software bugs, although a patch is already available	
Attack pattern	Tested	Exploitable
Exploiting known application vulnerabilities	YES	NO

[1] The report contains source code excerpts of freely available tools and exploits of third parties, where necessary

3.3 Standard Software

	Authentication issues	
	The web application provides insufficient means of authentication to protect its resources.	
Attack pattern	Tested	Exploitable
Bypassing authentication	YES	NO

	Authorization issues	
	An unauthenticated or unprivileged user is able to gain access to resources that are or should be protected.	
Attack pattern	Tested	Exploitable
Accessing protected functions	YES	NO
Accessing protected resources	YES	NO

	Business Logic issues*	
	The attacker is able to violate business rules of the application	
Attack pattern	Tested	Exploitable
Bypassing business rules	YES	NO

	Disclosure of information	
	The attacker is able to collect information about application internals or the server environment	
Attack pattern	Tested	Exploitable
Exploiting file extension handling	YES	NO
Gathering information from code comments	YES	NO
Gathering information from system- and error messages	YES	NO
Reading old, backup and unreferenced files	NO	NO

	Facilitation of client-side (web browser) attacks	
	This vulnerability class is web-related. It covers attacks that target the web browser.	
Attack pattern	Tested	Exploitable
Cross Site Request Forgery (XSRF)	YES	NO
HTML Injection / Cross Site Scripting (XSS)	YES	NO
HTTP Response Splitting / header injection	YES	NO
Frame Spoofing	YES	NO
Session fixation	YES	NO

	Interpreter injection / input validation problems	
	The application passes input parameters to the database, operating system APIs, or other interpreters without proper validation.	
Attack pattern	Tested	Exploitable
Accessing the file system	YES	NO
Code injection	YES	NO
Command injection	**YES**	**YES**
Format string injection	NO	NO
IMAP/SMTP injection	NO	NO
LDAP injection	YES	NO
ORM injection	NO	NO
Overflowing character buffers	NO	NO
Path traversal	YES	NO
SQL injection	YES	NO
SSI injection	NO	NO
XML injection	YES	NO
XPath injection	NO	NO

	State / session management problems	
	State- or session-variables are initialized and used incorrectly.	
Attack pattern	Tested	Exploitable
Enumerating session identifiers	YES	NO
Exploiting session state issues	**YES**	**YES**

	Unsafe management of trusted data	
	Trusted or application-internal data can be manipulated by the attacker	
Attack pattern	Tested	Exploitable
Manipulation of application-internal data on the client	NO	NO
Reading application-internal/confident data on the client	NO	NO

	Unneeded / unsafe functionality	
	The application provides inherently unsafe functionality	
Attack pattern	Tested	Exploitable
Exploiting sample applications	NO	NO
Upload of arbitrary files	YES	NO

	Unsafe algorithms	
	Use of unsafe algorithms allows compromise of sensitive data	
Attack pattern	Tested	Exploitable
Breaking encryption	YES	NO
Exploiting weak RNG	NO	NO

	Vulnerability to denial of service	
	The service can be rendered unusable by the attacker	
Attack pattern	Tested	Exploitable
Exploiting unlimited resource allocation	YES	NO
Locking customer accounts	NO	NO

* For items which are marked as **NO** in the **TESTED** column; the vulnerability is not practical to be tested due to its nature which is not compatible with the nature of the module tested. For example: No test will be performed to identify SQL injection on a static text file.

4 Severity Level Classification

All security risks discovered were evaluated with a risk score. The risk score is calculated from a risk matrix, which consists of likelihood and severity. The likelihood describes the probability that an attacker discovers the vulnerability and is able to exploit it. The severity refers to the severity of the vulnerability as well as its impact. As the severity influences the risk stronger than the likelihood, it is included squared in the equation. By multiplying likelihood and severity, the risk score is determined, which allows an assessment of the risks posed by a vulnerability.

Severity / Likelihood	1	4	9	16	25
1	1	4	9	16	25
2	2	8	18	32	50
3	3	12	27	48	75
4	4	16	36	64	100
5	5	20	45	80	125

To allow for a simple textual description of the risk, the scores were classified into four main categories:

Risk Score	Risk assessment
1 – 10	low
11 – 24	medium
25 – 60	high
61 – 125	critical

4.1 Definition of The Term Likelihood

The "likelihood" identifies the probability that the flaw can be exploited by an attacker. It is influenced by a combination of the following factors:

- **User Privileges Required / Network access required:**
  In general, the lower the privileges required by an adversary, the higher the likelihood of an exploit. However, this factor heavily depends on the defined attack scope and the audit goal, e.g. are we assuming that the attacker is already administrator or are we assuming that the attacker starts as an unauthenticated user.
- **User Interaction:**
  The fewer user interactions required (in UI) by the victim(s), the higher the likelihood of an exploitation by an adversary.

- **Attack Complexity / Time Required:**
  The lower the "attack complexity", the higher the likelihood of an exploit. This factor only decreases the likelihood notably if large resources (time/computing power) and / or very large samples of data (e.g. network traffic) are required for a successful exploit.
- **Existence of Public Exploits:**
  If exploits are available to the public (for free or via readily available commercial tools), the likelihood increases significantly.
- **Knowledge about System Internals:**
  The fewer knowledge required about the systems internals (e.g. access to configurations), the higher the likelihood of an exploit. This factor only decreases the likelihood notably, if the auditor has significantly more knowledge than the assumed attacker.
- **Chaining of Vulnerabilities:**
  In some cases, a vulnerability can only be fully leveraged when chained with other vulnerabilities. Based on the specific attack assumptions and other relevant (non-)existing vulnerabilities, the factor "Chaining of vulnerabilities" can increase or decrease the likelihood significantly in certain cases.

Depending on the specific flaw identified and the defined audit scope, certain factors may be weighted more than others.

Factors that are **not** considered for the likelihood of a flaw:

- **Skill level of attacker.**
  Not factored in. It describes the general competence of an attacker. We always assume that an attacker is at least as smart as a SEC Consult auditor.

4.2 Definition of The Term Severity

The term "severity" defines the impact of the identified flaw. The higher the severity, the higher the costs associated with a successful exploitation of the identified flaw by an adversary.

5 Findings

These tables below list out the findings identified throughout the assessment. The subsequent action/remark towards these findings can be seen below.

5.1 Critical Severity Findings

No.	Findings	Recommendation
No findings found		

5.2 High Severity Findings

No.	Findings	Recommendation	Status
No findings found			

5.3 Medium Severity Findings

No.	Findings	Recommendation	Status
1.	CSV Injection	Please refer to 6.1.2 for solution.	
2.	Unsafe Cookie Handling	Please refer to 6.2.2 for solution.	

5.4 Low Severity Findings

No.	Findings	Recommendation	Status
1.	Missing HTTP Security Header	Please refer to 6.3.2 for solution.	

6 Finding Details (Web Application)

6.1 CSV Injection

Severity: **Medium** (Severity: 9, Likelihood: 2)

6.1.1 Description

Certain parts of the application have a "CSV export" feature which allows the export of user influenced data as a CSV or XLS file. In those parts of the application, it is possible for an attacker to set certain values in the application that - when exported and opened with a spreadsheet application (Excel, Open Office, etc, ...) - will be interpreted as a formula. This type of attack is called CSV formula injection. It is also known as CSV Excel Macro Injection.

This is dangerous as it puts the user who opens those malicious exported files at risk. Exfiltration of sensitive data or even the execution of arbitrary code on the local machine of the victim will be the result. The final impact depends on the used spreadsheet software on the client of the victim. Due to the limited time frame a comprehensive check was not possible. It can be assumed that similar vulnerabilities exist in the application.

6.1.2 Solution

All fields that are included in a CSV or spreadsheet export and which can be influenced by the user must be validated or filtered accordingly. One solution would be, for example, to include all inputs with single quotes before export. This solution must be implemented for all user attributes that are included as part of an export.

Even though Microsoft Office and some other spreadsheet applications are aware of such attacks and notify user with warning messages before processing such injected files, it is very likely that users ignore such error messages, since the dangerous file originates from a trusted source; in this case from the vulnerable application itself.

More information can be found at:

https://www.owasp.org/index.php/CSV_Excel_Macro_Injection

6.1.3 Proof of Concept

The application does not validate the content of CSV file that is generated by the application. This allows a malicious user to inject spreadsheet formula that gets executed in the users system.

Following is payload that can be used to demonstrate this attack scenarios.

```
=cmd|'/C calc'!A0
```

To demonstrate the attack, the authenticated attacker creates a malicious csv injection through data entry in the global trade panel.

We will inject the CSV payload at the Custody Account Name field for custody account no of C14011410046.

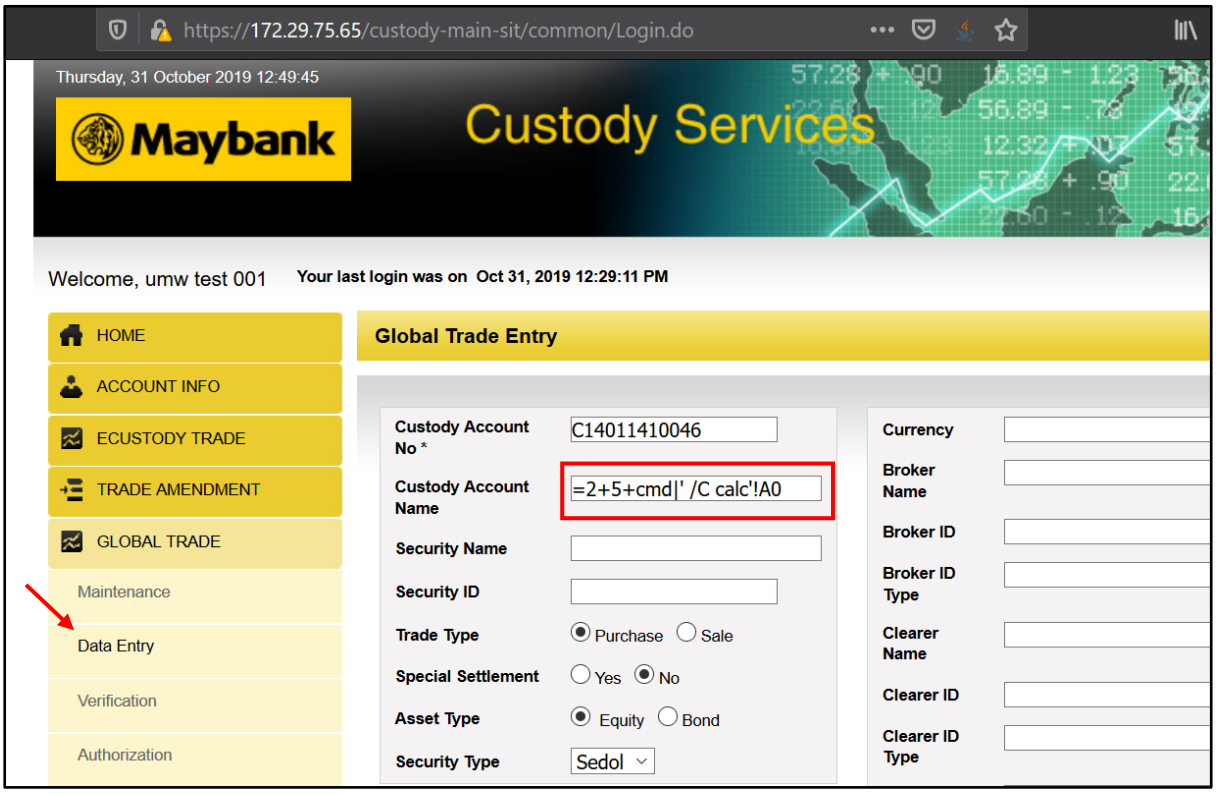

Figure 1: Injecting payload at the Custody Account Name field.

Now go to Maintenance page under the Global Trade panel and search for previous data entry, C14011410046. Click Export to CSV.

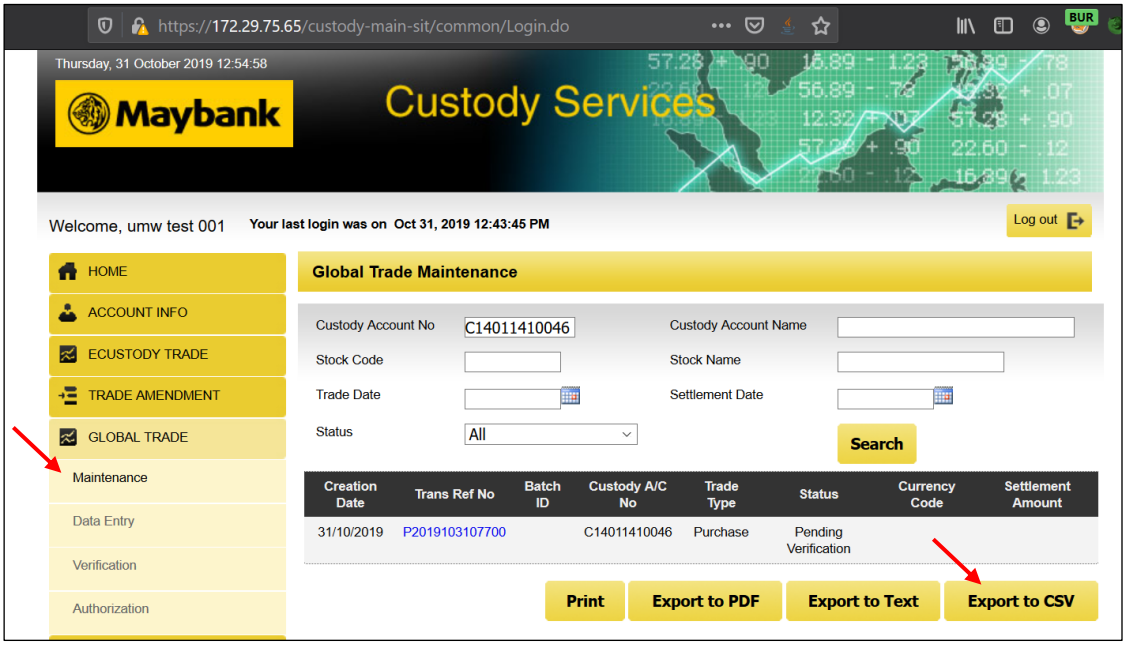

Figure 2: Converting the data into CSV.

As soon as the victim opens the file in any spreadsheet application, the payload (calculator) would get executed in the victims' system. Side note: warnings might pop-up when opening this csv file; However, these warnings are usually ignored because the file comes from a trusted source.

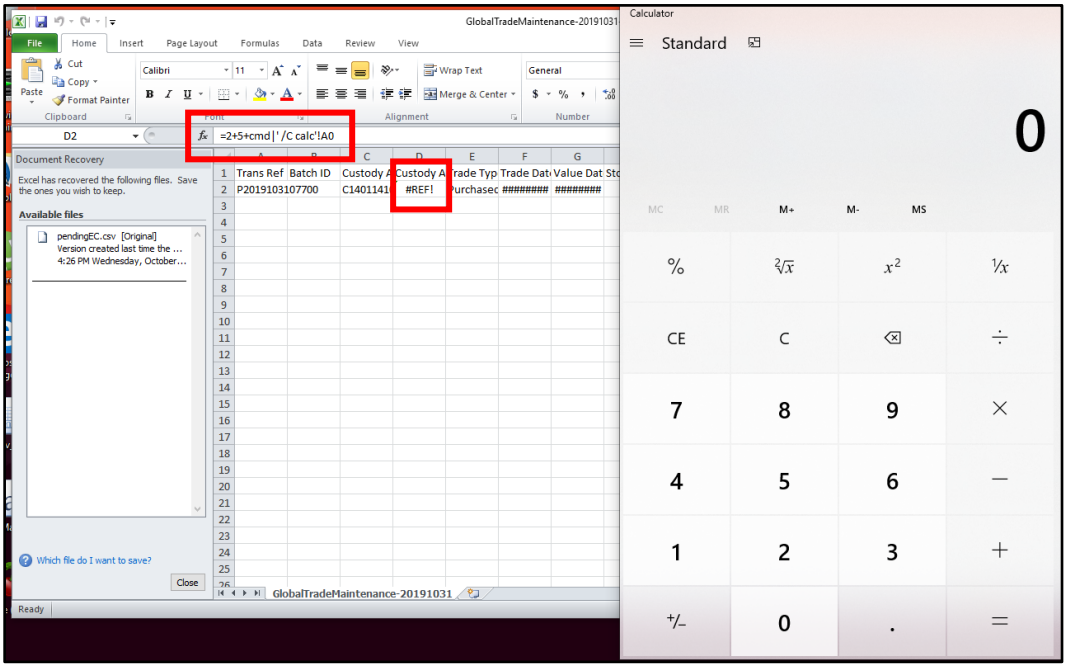

Figure 3: The payload was executed once the victim open the csv file.

6.2 Unsafe Cookie Handling

Severity: **Medium** (Severity: 9, Likelihood: 2)

6.2.1 Description

The application uses cookies in an unsecure way. If session cookies or other security-relevant cookies can be stolen by an attack, this often enables the attacker to takeover user sessions and user accounts.

The cookies are transmitted without the secure flag. This allows an unencrypted transfer of cookies back to the server. By setting the secure flag, the browser of the client is instructed to transmit the cookies to the server exclusively via encrypted connections. This makes it considerably more difficult to read the cookies in the course of man-in-the-middle attacks.

6.2.2 Solution

The secure flag should be enabled for the transmission of cookies. Also, recommended to change the value of the cookie with each and every request made.

6.2.3 Proof of Concept

The following GET request shows that the cookies of the website are not configured with the secure flag.

```
GET /custody-main-
sit/ib110/tradeGlobalMaintMain.do?SECONDARY_TOKEN=8c5cf15d44e0e477cb1c2ba99
7ed7cc8&nId=92 HTTP/1.1
Host: 172.29.75.65
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0)
Gecko/20100101 Firefox/70.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: https://172.29.75.65/custody-main-
sit/ib110/tradeGlobalMaintMain.do?SECONDARY_TOKEN=3911199ae077cc18c467ae7f7
461fb3e&nId=92
Cookie:
JSESSIONID=pV47d6qGBcvZqnmqbW9Y5Q37PCbT48LgPSV1RWxnNhNBVg12CNTn!1772184174
Upgrade-Insecure-Requests: 1
```

6.3 Missing HTTP Security Header

Severity: **Low** (Severity: 1, Likelihood: 5)

6.3.1 Description

The web application does not make use of security features that are available in modern web browsers. The security headers `X-XSS-Protection` (reflective Cross-Site Scripting protection), `Strict-Transport-Security` (protection against man-in-the-middle attacks), `Content-Security-Policy` (protection against Cross-Site Scripting and Clickjacking attacks) and `X-Content-Type-Options` (protection against MIME-type confusion) are not set by the application. Usage of these headers might stop attackers from exploiting various security vulnerabilities.

6.3.2 Solution

To safeguard end-users against known attack vectors the application should make use of optional HTTP security headers. Special security headers are one of the possible ways to enhance the existing security level. This might prevent certain attacks or makes their exploitation harder.

More information can be found at:

https://www.owasp.org/index.php/List_of_useful_HTTP_headers
https://www.owasp.org/index.php/Clickjacking_Defense_Cheat_Sheet
https://www.owasp.org/index.php/HTTP_Strict_Transport_Security
https://www.owasp.org/index.php/Content_Security_Policy

6.3.3 Proof of Concept

To verify this issue, an ordinary HTTP response from the webserver can be analyzed (no special manipulation of the request needed). We can see from the response below `X-XSS-Protection`, `Strict-Transport-Security`, `Content-Security-Policy` and `X-Content-Type-Options` are missing.

```
HTTP/1.1 200 OK
Date: Thu, 31 Oct 2019 05:06:40 GMT
X-frame-options: SAMEORIGIN
Cache-control: no-cache,private,no-store
Pragma: no-cache
Content-type: text/html; charset=ISO-8859-1
Expires: Wed, 31 Dec 1969 23:59:59 GMT
Connection: close
[/snip]
```

7 Version History

Version	Date	Status/Changes	Created by	Responsible
1.0	2019-10-31	Final Report	A. Zulkifli	W. Ikram