ASSIGNMENT-III

SECURE SOFTWARE ENGINEERING

Stack-Based Buffer Overflow

Stack-Based Buffer Overflow

Heap-based Buffer Overflow

Threat Modelling for online banking system

Submitted by,

Kishaiyan Vellaichamy Thangaraj

Part-1

Vulnerability-1

Stack-Based Buffer Overflow (CWE-121)

Link to the file- https://github.com/blender/blender/source/blender/blenkernel/intern/blendfile.cc

Link to the commit-
https://github.com/blender/blender/commit/1a04a231ec0ddec79189035310f3739b94f3339e#diff-8f360a798e7f019f3a3566b8f405631e776855a8f7325a041238fe172b5a11b1

Name of the file- blendfile.cc

The programming language used is C++.

Name of the repository- blender

Number of repository stars- 9.7k

Number of contributors in the repositories- 223

Type of Vulnerability CWE- 121

The vulnerability in the above file blendfile.cc is a type of buffer overflow called a stack-based buffer overflow. Initially the code strcpy() was to copy the copy the string from the source to the destination without checking the size of the copied string from the source. This casues the stack-based buffer overflow because there was no check for the size of the string that can be copied to the destination. The fix for this vulnerability is changed in lines 170 and 173 that is size_t that holds the maximum number of characters that can be copied to the destination and using strncpy() instead of strcpy() that checks for the maximum number of characters that can be copied will help in tackling the vulnerability. The fix uses a modified version of strncpy() method which is a standard mitigation technique for vulnerability with few extra safety features.

Vulnerability-2

Stack-Based Buffer Overflow (CWE-121)

Link to the file-
https://github.com/Bforartists/Bforartists/source/blender/blenlib/intern/path_util.c

Link to the commit-
https://github.com/Bforartists/Bforartists/commit/25aa510adbdc2d3c769cd8d8645fcbf9a9cf1077

Name of the file- path_util.c

The programming language used is C++.

Name of the repository- Bforartists

Number of repository stars- 2.6k

Number of contributors in the repositories- 88

Type of Vulnerability CWE- 121

The vulnerability found in the file path_util.c is a type of buffer overflow known as Stack-Based Buffer Overflow. There is a potential buffer overflow vulnerability in the code because of the maximum number of characters that can be copied from source to destination. As it does not check for the maximum number of characters instead copies everything including name_end introduces the buffer overflow vulnerability that can be used to exploit and access sensitive information. The fix in this scenario is checking the limit of characters so that the vulnerability cannot be exploited. They have fixed the vulnerability by using MIN2 function that selects the minimum by comparing the maximum number of characters, and everything copied including name_end. The standard mitigation technique for this vulnerability is to use strncpy() which has a vulnerability if the checking number is not defined properly. Therefore, there is another condition that selects the minimum from comparing the two.

Vulnerability-3

Heap-Based Buffer Overflow (CWE-122)

Link to the file- https://github.com/assimp/assimp/code/AssetLib/Obj/ObjFileImporter.cpp

Link to the commit-
https://github.com/assimp/assimp/commit/54f5d01190d0952c5d4fcae1f34d7e3bba0a14e5

Name of the file- ObjFileImporter.cpp

The programming language used is C++.

Name of the repository- Assimp

Number of repository stars- 9.6k

Number of contributors in the repositories- 548

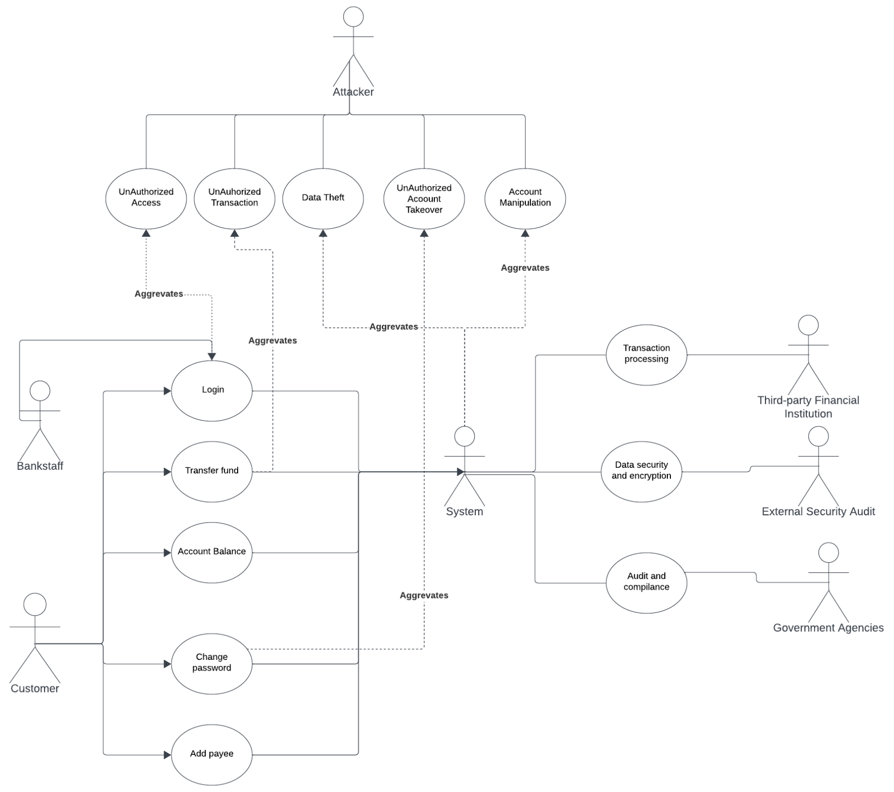
Type of Vulnerability CWE- 122

The ObjFIleImporter.cpp file has a type of buffer overflow known as the Heap-Based Buffer Overflow. The heap-based buffer overflow is caused by incorrect or unchecked write conditions for dynamically allocated memory buffer, they usually involve improper memory management, incorrect indexing. The vulnerable code had manual dynamically allocated memory and the caller is responsible for releasing the memory which is not a proper way of managing the memory leading to heap-based buffer overflow. This can be fixed by using proper memory management like std::unique_ptr that takes the ownership of the memory allocation and memory is released when it goes out of scope, making it secured method of handling the memory. It is a standard mitigation technique for improper memory management, one cause of heap-based buffer overflow.

Part-II

Software System:

An online banking system, often referred to as internet banking or e-banking, is a digital platform provided by banks and financial institutions to offer customers the ability to conduct a wide range of financial transactions and account management tasks over the internet. It provides a convenient and secure way for users to access and manage their financial accounts, conduct transactions, and obtain banking services without the need to visit a physical branch.



Description of (Mis)use cases:

1.Login (Unauthorized Access):

An Attacker tries to gain unauthorized access to the user’s account by repeatedly entering password or hijacks the session.

2.Transfer Fun (Unauthorized transaction):

An Attacker tries to transfer funds to another account through the vulnerability.

3.View Account Details (Data Theft):

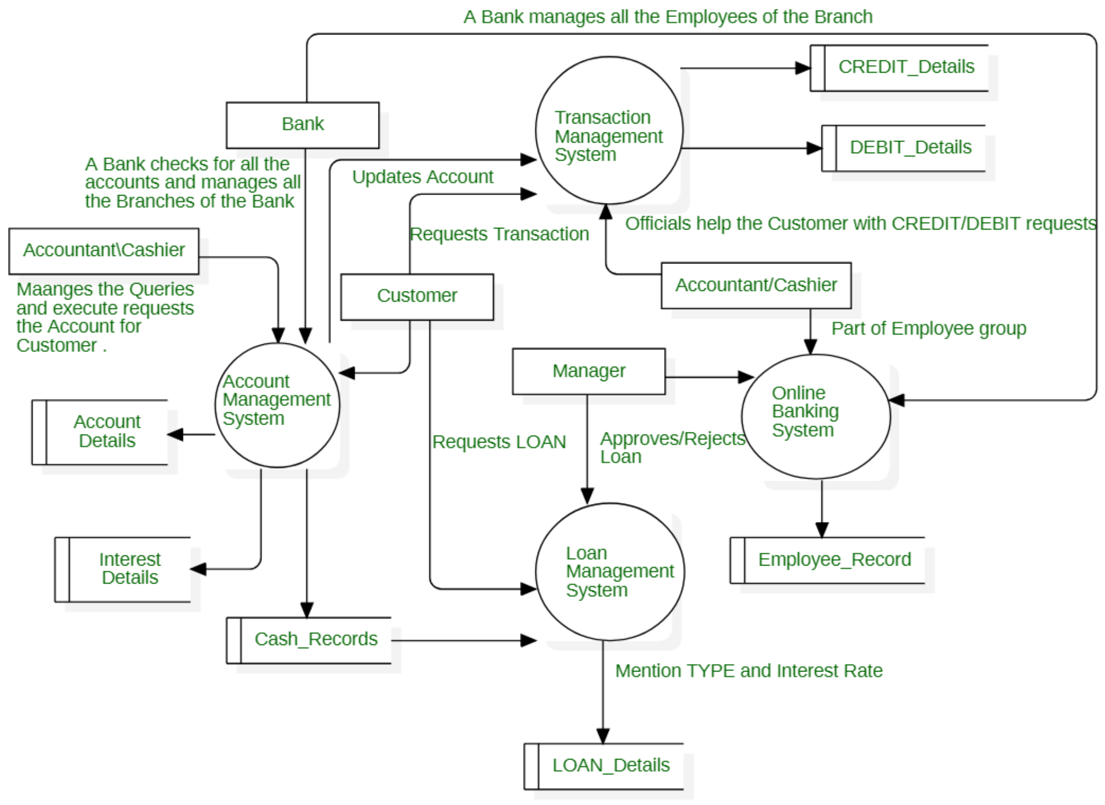
An Attacker tries to view the details of the customer/user without authorization through a vulnerability in the system.

4.Change Password (Unauthorized account takeover):

An Attacker tries to change the password for the account to gain access to the user’s account potentially leading to unauthorized account takeover.

5.Add payee (Account Manipulation):

An Attacker tries to add a fraudulent payee to the user’s account for malicious purposes.


A Bank manages all the Employees of the Branch
CREDIT_Details
Bank
Transaction Management System
DEBIT_Details
A Bank checks for all the accounts and manages all the Branches of the Bank
Updates Account
Requests Transaction
Officials help the Customer with CREDIT/DEBIT requests
Accountant\Cashier
Maanges the Queries and execute requests the Account for Customer .
Customer
Accountant/Cashier
Part of Employee group
Account Management System
Manager
Online Banking System
Account Details
Requests LOAN
Approves/Rejects Loan
Interest Details
Loan Management System
Employee_Record
Cash_Records
Mention TYPE and Interest Rate
LOAN_Details

LEVEL-1

Security Threats:

Unauthorized Access:

This threat occurs when an attacker attempts to gain unauthorized access to the system's database, potentially through SQL injection or other means. Unauthorized access can lead to data theft, manipulation, or even complete system compromise. The risk of this security threat is 5.7 based on the calculation done by CVSS version 3.0 meaning that it is moderate threat.

Mitigation: The inputs from the user can be sanitized and the special characters can be escaped for tackling SQL injection.

Data Interception:

Attackers may attempt to intercept data flows, such as login credentials, transaction details, or account balances, while they are in transit between the user's device and the online banking system. This can lead to sensitive data theft and account compromise. The risk for this security threat is 6.1 based on the calculation done by CVSS version 3.0 indicating this as a moderate threat.

Mitigation: Use encryption (e.g., SSL/TLS) to secure data in transit, implement secure session management, and educate users about safe browsing practices to avoid phishing.

Identity Spoofing:

Attackers may attempt to impersonate users by stealing their session tokens or cookies. This can lead to unauthorized access to user accounts and fraudulent transactions. This security threat poses a risk of 6.6 on CVSS 3.0, higher than the other two security threats and should have the priority to be fixed.

Mitigation: Implement secure session management, use secure cookies, and regularly rotate session tokens. Employ two-factor authentication for added security.