Secure Software Engineering Lab

Lab 2: Secure Software Design

Institute of Software Security (E-22)

1. Objectives

Apply the knowledge acquired in the lectures on the following areas of software security:

- Tactics for secure software design.
- Security patterns.
- Secure software architectures.

2. Tasks

1. **Analyze each of the security requirements.**
2. **Identify and discuss suitable security tactics for each requirement**.
3. **Identify, instantiate, and discuss suitable security patterns for each requirement.**
   - Browse the security patterns catalog.
   - Discuss in pairs.
   - Instantiate the pattern.
4. **Discuss the benefits and limitation of each selected pattern.**

3. Materials

Case study, lecture slides, lab slides, security requirements, security patterns catalog.

Summer Semester 2022

Appendix - Security Requirements

Security Requirement #A

Context: Decentraland is a 3D metaverse platform where people can create avatars to represent themselves and take part in real life activities like socializing, attending concerts or buying assets. People can pay for various items and services using virtual or crypto currency. There is a virtual bank called Meta Wallet, set up to enable easy transactions of crypto currency. The bank clerks of Meta Wallet use a variety of systems to perform their work, including the internal virtual bank application. They also use services provided by a Bank Token company (that offers services like blocking a token to prevent it from retrieving cryptocurrency), and by Fraud Detection Company (to detect if an avatar is validated and track the transaction patterns). For the internal service, the clerk uses an account issued by Meta Wallet itself, while for the external services she uses the accounts issued by each third party. This situation implies that a bank clerk has at least three different credentials to authenticate to the above services. As a security measure, the security policy of the bank prescribes rigid password complexity rules and regular password changes. Furthermore, it is forbidden to use the same password among multiple, separate systems. Most importantly, a password for an internal service may not be reused to access any external system. Sharing passwords would make the security of the virtual banking system depend on the security practices of external companies, and potentially give external parties access to the internal passwords of bank employees. Of course, this situation is to be avoided. The obvious solution, namely requiring unique, complex, and regularly changing passwords for each of these services would be cumbersome for the bank employee. Some services are not used on a daily basis, and remembering the password thus becomes difficult. Therefore, chances are high that the employee would resort to writing down the various passwords.

Security Requirements: The system should force the bank employee to comply with the bank's security policy (using strong passwords that are not shared between services), but at the same time reduce the number of passwords that a bank clerk has to remember, possibly to only one. This is often referred to as single sign-on (SSO). Furthermore, possible evolution towards stronger authentication mechanisms (e.g., using a virtual smart card next to a password) should be taken into account.

Additional Constraints:

- The interfaces of the external parties (the Bank Token Company and the Fraud Detection Company) are fixed, and cannot be modified.

Security Requirement #B

Context: Meta Wallet has introduced a mobile application which can be accessed via metaverse phones or meta watches, with which customers can access their wallet information without having to visit the bank in Decentraland. It includes the balance in their wallets and the most recent transactions for each wallet. They can also perform crypto currency transfers with the mobile application. For security reasons, however, these transfers are limited to beneficiaries that are stored in a contact list. This contact list is created through the Meta Wallet Online interface (and cannot be modified from within the mobile application). Meta Wallet customers can enjoy the functionality of their mobile application everywhere even if they are not present in Decentraland. Therefore, the mobile application caches customer data in their database. When the customers are not present in Decentraland, the most recent information may not be available and the application will use the cached data. In this way, the customers can still consult their latest known account balances and recent transactions. This information, however, is clearly labeled as 'possibly outdated'. Also, the customer can initiate a cryptocurrency transaction outside Decentraland; in that case, the transaction is executed as soon as they enter Decentraland again.
The cached information includes the last known wallet balances, recent transactions, the approved contact list and the pending transactions. Moreover, when the application is started, the customer has to enter an application-specific PIN code to avoid unauthorized use. The validation of the PIN code should be performed locally on the device, so that the application can be used outside of Decentraland. The mobile application also locally stores the credentials that are used to authenticate towards the Meta Wallet system. These credentials are only used after the correct PIN code has been entered.

Security Requirements: The information that is stored on the device is security-sensitive, and should be protected against disclosure to unauthorized parties. Also, the mobile application should be the only entity entitled to alter the stored information.

Additional Constraints:

- The mobile application runs on the Android 11.1 (or newer) platform. This platform offers an API for accessing both an application-only storage area and a shared storage (e.g., an SD card). The application-only storage area is not accessible to other applications (if the device is not rooted).

Security Requirement #C

Context: Clearly, the Meta Wallet system needs a publicly available web server to enable an online interface for the customers. This server receives requests from unauthenticated origins that come from everywhere, for instance customers that want to log in into the system. There is a similar service for the mobile customers, offering the API that is used by the mobile application. This is a sensitive server, as it provides a link between the outside, possibly hostile world (Decentraland) and the internal, critical systems of the bank. Therefore, the bank's system consists of a less critical part (e.g., the web server) and a critical part (e.g., the transaction management). Indeed, the transaction store is a crucial element for the correct working of the entire Meta Wallet, and should be shielded from possible attacks from outside at all cost. On the other hand, the web server (although important for the customers) is less critical for the core operation of Meta Wallet. Malicious traffic should therefore not be able to pierce the trust boundary between these two parts. Nevertheless, the web server needs to be able to connect to the internal subsystems in order to provide its functionality to the customers.

Security Requirements: A security vulnerability in a publicly available system (e.g., on the bank's web server software) should not allow an outside attacker to obtain direct read access to the information in the transaction store.

Additional Constraints:

- Pay attention to the fact that load balancing is used for the web server, to accommodate for a possibly large number of simultaneous requests.
- Focus on unauthenticated outsiders as attackers, i.e., not customers of the system.

Security Requirement #D

Context: In an average usage session, a customer of the Meta Wallet mobile application typically performs a series of operations in close sequence, for example checking his balance before performing a number of payments (transfers). In the future there will be ways to authenticate users through eye recognition or facial scanning. However for the time being, authentication is done via credentials. Clearly, it is infeasible to require the customer to re-enter his credentials every time a request is made. Nevertheless, the system needs to be able to identify the originator of each request.
To perform multiple operations in sequence, some customer-specific data needs to be kept. For example, customers can enter multiple transfers in a row, before confirming them all at the same time. This requires that the data about the already entered transfers is temporarily stored and associated with the corresponding customer. When a subsequent request arrives, this data is retrieved and updated if necessary.

Security Requirements: Customers that access Meta Wallet via the online interface should not have to re-authenticate for every request that they make, but only once. As the customer does not explicitly authenticate for every request, care must be taken that the customer-specific data is never mixed up with the data of other customers while interacting with Meta Wallet online interface.

Additional Constraints:

- There should also be a possibility for the customer to notify the system that he is done using it.
- Pay attention to the fact that load balancing is used for the web server, to accommodate for a possibly large number of simultaneous requests.

Security Requirement #E

Context: The banking industry is highly regulated to prevent misbehavior and fraud, also from insiders. For example, in the United States, banks must comply with the Sarbanes-Oxley Act (SOX). One aspect of SOX and similar regulations relates to preventing manipulation, destruction or alteration of financial records or other interference with investigations. The regulations also provide some protection for whistle-blowers, i.e., a person reporting to an authority about dishonest or illegal activities.
In particular, it should not be possible for a bank employee to take advantage of her function in the bank. For example, a bank employee should not grant a loan to herself, or use information obtained via her position in the bank to provide investment advice to family and friends. In particular, in the Meta Wallet system it should be possible to identify the clerk that has performed each action related to a customer's account, for example when a transfer is performed on behalf of the customer. This evidence is used by the bank in case of a dispute with a customer, in order to verify the conduct of the involved parties. This information is also used by the bank to routinely monitor the behavior of the employees to serve as a deterrent for any misconduct.

Security Requirements: All actions that involve a customer's crypto wallet, including reading the balance or performing a transaction, that are performed by a bank clerk should be tracked by the system. It should be possible to detect misbehavior of a bank clerk, and the cleck should not be able to plausibly deny his involvement in any misconduct.

Additional Constraints:

- Meta Wallet is not allowed to rely on external services to accomplish this requirement.
- Bank clerks cannot object to their actions being monitored by Meta Wallet

Security Requirement #F

Context: Due to the nature of the bank's business domain, customers may be tempted to commit fraud in order to earn easy crypto currencies. This can be obtained by providing carefully crafted inputs to the bank system (via one of the channels available to the customers), so that the bank system is tricked into transferring the crypto currency to the fraudulent customer's wallet. Due to the sensitive nature of financial information, some people could also be motivated to try to obtain such information about the bank's customers. This is also true for existing customers of the bank, which for instance may be tempted to learn about the financial standing of a VIP. If these attempts would be successful and revealed to the general public, the reputation of the bank would suffer severely.

Security Requirements: The system should let customers perform operations on their own accounts only. Examples of such operations are checking the account balance, checking the transaction history, or transferring an amount.

Additional Constraints:

- Focus on authenticated customers trying to misbehave.
- Do not focus on the fraudulent use of bank tokens.

Security Requirement #G

Context: Metaverse phones, smartphones or meta watches are often connected via a public wireless network. Because some of these networks can be easily controlled by a malicious user, the data that is sent over this network should be protected.
The data that is exchanged between the bank and the mobile application (e.g., wallet balances, transactions, etc.) is sensitive in nature. Therefore, it should only be exchanged between and known to legitimate parties. Also, it is important that the data is not altered while it is being exchanged, so that the same data that is sent from the mobile application arrives at the bank (e.g., transfers issued by the customer using the mobile application), and vice versa (e.g., the list of beneficiaries to whom the customer is allowed to issue transfers via the mobile application).

Security Requirements: All data that is transmitted between the bank and the customer's mobile device (and vice versa) must be protected against unauthorized disclosure and modification.

Additional Constraints:

- An authentication mechanism of the mobile client (customer) towards the bank system is already in place.