DOP: / /2023 **DOS: / /2023**

Experiment No:3

Title: Modeling Threats in android using STRIDE

Theory:

- **Modeling Threats:**

Threats can be anything that can **take advantage of a vulnerability to breach security** and negatively alter, erase, harm objects or objects of interest. Threat Modelling can be done at any stage of development but if done at the beginning it will help in the early determination of threats that can be dealt with properly.

The **purpose of Threat modelling** is to identify, communicate, and understand threats and mitigation to the organisation's stakeholders as early as possible. Documentation from this process provides system analysts and defenders with a complete analysis of probable attacker's profiles, the most likely attack vectors, and the assets most desired by the attacker.

Threat modelling helps to achieve the following:

- Defines security of application
- Identifies and investigates potential threats and vulnerabilities
- Results in finding architecture bugs earlier.

STRIDE:

STRIDE is a methodology developed by **Microsoft for threat modelling**. It provides a mnemonic for security threats in six categories:

- **Spoofing Identity:**

Identify spoofing occurs when the hacker pretends to be another person, assuming the identity and information in that identity to commit fraud. A very common example of this threat is when an email is sent from a false email address, appearing to be someone else (also called a phishing attack). Typically, these emails request sensitive data. A vulnerable or unaware recipient provides the requested data and the hacker is then easily able to assume the new identity.

- **Tampering With Data:**

Data tampering occurs when data or information is changed without authorization. Ways that a bad actor can execute tampering could be through changing a configuration file to gain system control, inserting a malicious file, or deleting/modifying a log file.

- **Repudiation Threats:**

Repudiation threats happen when a bad actor performs an illegal or malicious operation in a system and then denies their involvement with the attack. In these attacks, the system lacks the ability to actually trace the malicious activity to identify a hacker.

Repudiation attacks are relatively easy to execute on e-mail systems, as very few systems check outbound mail for validity. Most of these attacks begin as access attacks.

- **Information Disclosure:**

Information disclosure is also known as information leakage. It happens when an application or website unintentionally reveals data to unauthorized users. This type of threat can affect the process, data flow and data storage in an application. Some examples of information disclosure include unintentional access to source code files via temporary backups, unnecessary exposure of sensitive information such as credit card numbers, and revealing database information in error messages.

- **Denial of Service (DoS):**

Occurs when an adversary uses illegitimate means to assume a trust level than he currently has with different privileges. Denial of Service (DoS) attacks restrict an authorized user from accessing resources that they should be able to access. This affects the process, data flow and data storage in an application. DoS attacks are getting bigger and more frequent, with an estimated 12.5 million DoS weapons detected in 2020.

- **Elevation of Privileges:**

Through the elevation of privileges, an authorized or unauthorized user in the system can gain access to other information that they are not authorized to see. An example of this attack could be as simple as a missed authorization check, or even elevation through data tampering where the attacker modifies the disk or memory to execute non-authorized commands.

- **Example Business Case**

The example business case, for our STRIDE threat modeling example, consists of:

Company and industry: Health care insurance provider, within the health care and insurance industry.

Object in the scope of threat modeling: The company is in the process of developing a new web application called 'MyHealth'. It will allow customers to view their medical data, billing, appointments with healthcare providers, and potential deals that customers can purchase.

Scoping limitations: Currently, the business processes related to the MyHealth application are not in scope.

Tech stack of the application: The MyHealth web application will be cloud-based, hosted in Azure, and will have a JavaScript framework front-end.

Team: Various DevOps teams are involved in the project, as well as business driving the transformation to provide as many health insurance services through the newly developed MyHealth application.

- **First Step: Gather Background Information:**

The first step in the STRIDE threat modeling example is to gather as much background information as possible (which sounds obvious but is still worth mentioning).

Potential information that may help in later STRIDE threat modeling steps:

Architectural diagrams and overviews of the proposed application design: Such architectural diagrams can provide threat modelers and team members with an understanding of key components within the design, technology used, communication flows, etc.

Requirements of the solution: Requirements often contain the business and IT requirements that explain what the application must be capable of doing. The amount and quality of requirements may vary in quality from project to project. There may even be

some security requirements that can help with the later STRIDE threat modeling steps.

Project documentation: Project documentation will explain how the application will be built, who is involved, how long the project will take, etc. Pay attention to the inclusion of security as part of the project. This will indicate whether security is an integral part of the project (and thus resulting application), or whether it is a 'forgotten' quality. Team overview and governance: Understanding the key players in a project is very important. Pay attention to the security team members involved (i.e., security champions, security department representatives, etc.).

- **Second Step: Create a Data Flow Diagram (DFD):**

Because we're applying the component-based STRIDE threat modeling method, we need a good understanding of the key components involved in the project, and how they interact with each other.

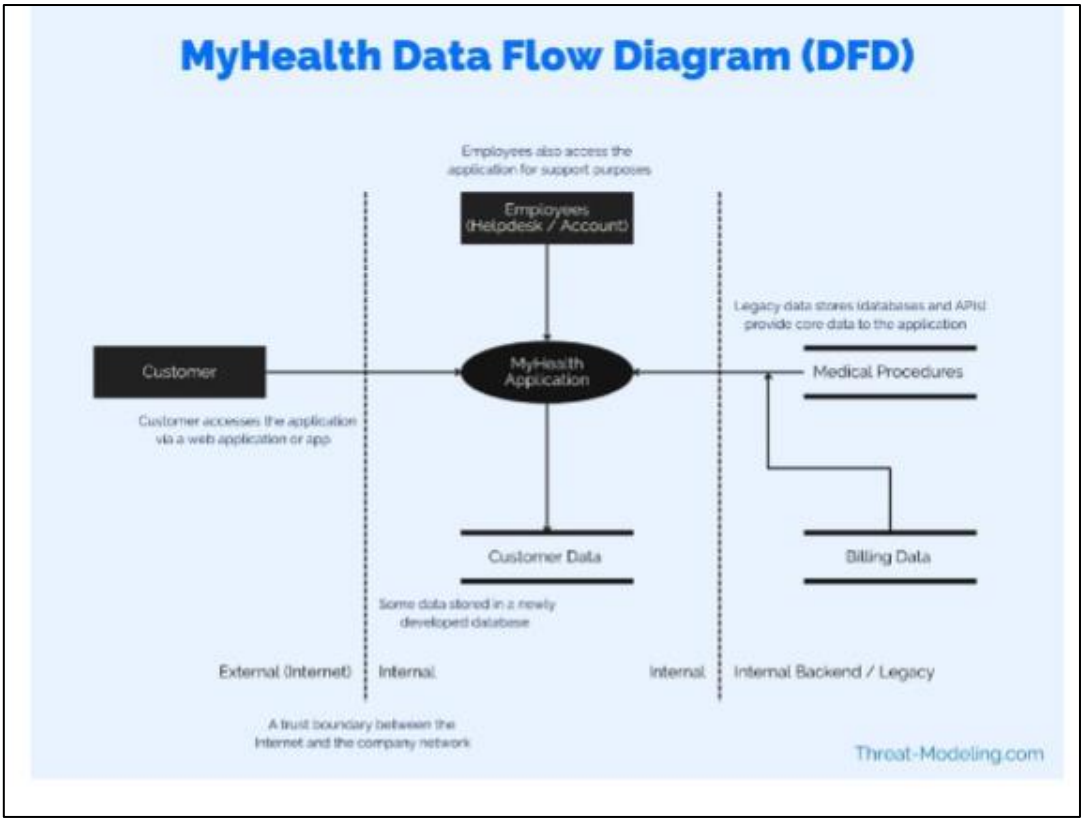
To achieve this, I'll create a Data Flow Diagram (DFD), which is the second step in the STRIDE threat modeling example. It will include only the key components, the key interactors (e.g., users), data stores, and the communication flows between them.

Architectural diagrams and overviews of the proposed application design help in creating a DFD.

The reason why I'm only including key components (versus including all components) is that including too much information can make the overall threat model too complicated and too difficult to understand. Participants in the threat modeling sessions will not be invested or engaged if they do not understand what they are threat modeling.

The Data Flow Diagram (DFD) should be made based on input from team members in an interactive threat modeling session. It is important to get input and cooperation from a wide range of team members, including technical/non-technical, developers/non-developers, testers, Product Owners, and any other people and roles with knowledge of the application.

The diagram below is an example of a Data Flow Diagram (DFD) based on the MyHealth example.



Customer: The end customer using the application. The customer is a health insurance recipient (paying a monthly health insurance fee, and receiving health insurance coverage).

Employees: Employees access the application as well as customers. Employees provide support to customers via a helpdesk and other channels. Employees can see more data than customers, including access to many different customers.

MyHealth Application: This is the main application in the scope of the threat modeling exercise. It is shown here as one icon. However, the application has many sub- components, backend, frontend, etc.

Customer Data: The new application will have a new database environment to store some of the relevant customer data.

Medical Procedures (and activities): Data provided to the customer includes medical procedures, activities, etc. This data is stored in a legacy database and system. This data will not be sent to the Customer Data database.

Billing Data: Billing data is managed in a separate system, but is shown via the MyHealth application (to the customer).

Note that the above Data Flow Diagram is not perfect, but for the purposes of keeping the STRIDE threat modeling example simple, it will do for now. Future posts will focus on how to tweak and improve on details included in a Data Flow Diagram.

- **Third Step: Perform Component Based STRIDE Threat Modeling:**

Now, as a result of creating a Data Flow Diagram, we have an understanding of the main components in the STRIDE threat modeling example. As a next step, I will perform STRIDE threat modeling against each component (known as component-based STRIDE threat modeling).

Two approaches for this step:

1. Think of potential threats per component, and assign a STRIDE threat type (i.e., assign Spoofing, or Tampering, etc.).

2. Per component, think of Spoofing threats specifically, then Tampering threats, then Repudiation threats, etc. So in order of STRIDE, go through the list and think of potential threats limited to the STRIDE threat type. I usually use both methods, or a combination of both.

For this example, I'll use method 2 – the method of strictly thinking per component, and walking through all STRIDE threat types, to determine which threats may apply.

Conclusion: -

Hence successfully performed Modeling Threat in Android using Stride.