Project Report: Threat Modeling for Sneaker Company App

Framework Used: PASTA (Process for Attack Simulation and Threat Analysis)

Author: *Mohanad Dalol*

Date: *29/07/2025*

Overview

This report documents the threat modeling activity performed for a sneaker trading mobile application using the PASTA framework. The app enables users to buy and sell sneakers, store sensitive data, and handle financial transactions.

Stage I: Define Business and Security Objectives

The sneaker app was developed with the following objectives:

- Enable seamless buying and selling of sneakers through an online marketplace.
- Ensure secure user registration, login, and account management.
- Comply with data privacy and financial transaction regulations (e.g., PCI-DSS, GDPR).

Stage II: Define the Technical Scope

Technologies Used:

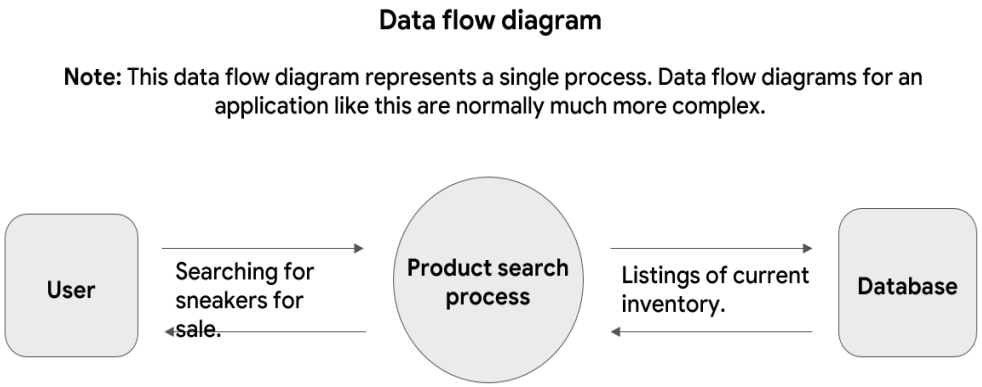
- **API:** Enables communication between mobile frontend and backend services.
- **PKI:** Used for secure data exchanges through digital certificates.
- **AES:** Encrypts user payment and profile data.
- **SHA-256:** Hashes passwords and sensitive fields.
- **SQL:** Manages database interactions for users, products, and transactions.

Justification:
SQL is prioritized as it handles all database operations including search, user login, and transactions, making it a critical attack surface. APIs are also prioritized due to their exposure to external clients and handling of sensitive endpoints.

Stage III: Decompose the Application

The following data flow was analyzed:

- **User** initiates a **search** request.
- Request is handled by **Product Search Process**.
- App queries the **Database** for inventory and returns listings.



This flow exposes sensitive operations (like data fetching) that must be secured.

Stage IV: Threat Analysis

Internal Threats:

- Misconfigured API endpoints could expose data.
- Developers may leave debug functions accessible in production.

External Threats:

- Attackers could inject SQL code into input fields.
- Brute-force attacks on login functionality could lead to credential compromise.

Stage V: Vulnerability Analysis

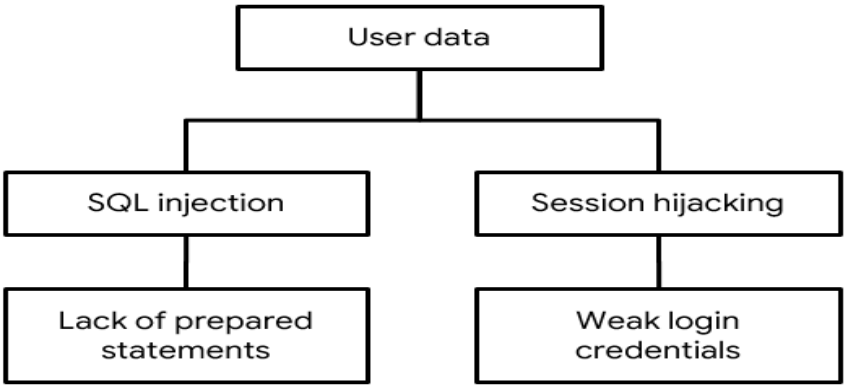
Key Vulnerabilities Identified:

- **SQL Injection:** Due to unsanitized input in database queries.

- **Weak Encryption:** Payment forms may not securely store or transmit credit card information.

Stage VI: Attack Modeling

Attack tree analysis revealed the following exploit paths:



This highlights the need to strengthen authentication and input handling.

Stage VII: Risk Analysis and Mitigation

Security Controls to Mitigate Risks:

1. **Prepared Statements:** Prevent SQL injection by enforcing parameterized queries.
2. **Strong Authentication:** Enforce password complexity and use multi-factor authentication.
3. **Transport Security:** Use TLS + AES for all data in transit.
4. **Input Validation:** Sanitize and validate all user-provided input.