Countermeasures Challenge for CycleSwift

Supporting materials for the O'Reilly training Threat Modeling Fundamentals - Debug Your Security Design through Whiteboard Hacking by Sebastien Deleersnyder, Toreon.

To learn more on threat modeling, visit https://www.toreon.com/threat-modeling-training/

Exercise Input:

Countermeasures exercise focusing on the payment integration of the CycleSwift E-Bike Rental App.

Hands-on Exercise: Prioritize Countermeasures for CycleSwift Payment Integration

Objective: Assess potential vulnerabilities in the payment integration of the CycleSwift app, prioritize them based on risk, and recommend effective countermeasures.
Potential Vulnerabilities in Payment Integration

1. **Insecure API Communication:**
    - Description: API endpoints used for payment processing might not be adequately secured, allowing for interception or manipulation of payment data.
    - Potential Impact: Financial loss, unauthorized transactions.
2. **Insufficient Authentication and Authorization:**
    - Description: Weak authentication mechanisms or flawed authorization checks could allow unauthorized users to initiate or manipulate payments.
    - Potential Impact: Unauthorized access, financial fraud.
3. **Storage of Sensitive Payment Information:**
    - Description: Sensitive payment information, such as credit card numbers, might be stored without proper encryption or in violation of compliance standards like PCI DSS.
    - Potential Impact: Data breaches, compromised user financial data.
4. **Lack of Payment Gateway Security:**
    - Description: Integration with the payment gateway might not follow best security practices, such as secure tokenization or proper validation of transaction responses.
    - Potential Impact: Man-in-the-middle attacks, payment fraud.
5. **Insecure Error Handling and Logging:**
    - Description: Verbose error messages or sensitive information logged improperly could expose details about the backend payment processing system to potential attackers.
    - Potential Impact: Information disclosure, aiding further attacks.
6. **Cross-Site Scripting (XSS) and Injection Flaws:**
    - Description: Payment forms or API endpoints might be susceptible to XSS or SQL injection, allowing attackers to inject malicious scripts or queries.
    - Potential Impact: Data theft, session hijacking, unauthorized transactions.

Challenge:

1. **Risk Ranking:**
    - Assess each vulnerability for its likelihood and potential impact. Consider factors such as the complexity of exploitation, potential financial loss, and damage to reputation.
    - Rank the vulnerabilities from highest to lowest risk.
2. **Recommend Countermeasures:**
    - For each vulnerability, propose specific countermeasures or security controls that would mitigate the risk.
    - Consider industry best practices, compliance requirements (e.g., PCI DSS for payment processing), and AWS-specific security features.
3. **Documentation:**
    - Describe your risk assessment and recommended countermeasures in a structured format.
    - Provide a rationale for each countermeasure, explaining how it addresses the vulnerability and reduces the risk.

Deliverables:

- A ranked list of the identified vulnerabilities based on their assessed risk.
- Detailed countermeasures for each vulnerability, including implementation suggestions or references to best practices.