ITCS411- Course Project

Project: Choose on of the following ideas:

1- Idea 1: Secure Chat Application *[Team size: up to 6 students]*
2- Idea: Network Traffic Anomaly Detection App and Attack Simulation App *[team size: up to 12 students].*

Deliverables:

1. Project Documentation: Write 3-7 technical report that involves the following:
    a. Provide a detailed project report that includes the design architecture, security mechanism and algorithms used, implementation details, and any challenges faced during development.
    b. User Guide: Create a user guide that explains how to set up and use the application, including screenshots.
2. Source Code: Submit the complete source code of the application, including all the necessary libraries and dependencies.
3. Presentation: Prepare a 5-minutes presentation to demonstrate the features and functionality of your project, highlighting the security measures implemented and discussing potential future enhancements.

Project idea 1: Secure Chat Application

Team Size [up to 6 students]

Description: Design and implement a secure chat application in python that allows users to communicate securely over a network. The application should incorporate cryptographic algorithms, SSL/TLS protocols, and robust security measures to protect the confidentiality and integrity of the messages exchanged between users.

Project Requirements:

1. User Registration: Implement a secure user registration process where users can create an account with a unique username and password. Use secure hashing algorithms to store and verify passwords securely.
2. Encryption: Implement end-to-end encryption for the messages exchanged between users. Use symmetric encryption algorithms (AES-256) to encrypt the messages and asymmetric encryption algorithms (RSA) to securely exchange and establish the session keys.
3. Link Scanning: Implement a link scanning module that analyzes the URLs shared in the chat messages. Use URL reputation services or APIs to check if the link is associated with known malware or malicious websites. Services like Google Safe Browsing API or VirusTotal provide URL scanning capabilities.
4. SSL/TLS Integration(optional): Integrate SSL/TLS protocols into the communication channels to provide secure connections between the client and server. Use libraries such as OpenSSL or PyOpenSSL to handle SSL/TLS certificates, key exchange, and secure communication.
5. User Interface (optional): Design a user-friendly interface for the chat application that allows users to send and receive encrypted messages. Provide features such as message history, contact lists, and real-time message updates.

Utilize network programming concepts to establish communication channels between clients and the server. Implement socket programming using Python's socket library or other suitable networking libraries.

Project Idea 2: Network Traffic Anomaly Detection App and Attack Simulation App

Team Size [up to 12 students] [Programming language: Python]

This project will provide the team with practical experience in building a robust detection system and evaluating its effectiveness under simulated attack scenarios. This project will enhance their understanding of network security, anomaly detection techniques, and the challenges involved in identifying and mitigating potential threats.

The team should build both the network traffic anomaly detection app and the attack simulation app.

Network Traffic Anomaly Detection App

Description: Develop a network traffic anomaly detection app that monitors and filters network traffic in real-time, aiming to identify abnormal behavior indicative of potential security threats. The app will analyze network packets, traffic patterns, and behavior to detect and raise alerts for suspicious activities.

Features:

1. Network Traffic Capture: Implement a module to capture and analyze network packets from the network interface. Utilize appropriate libraries or tools (e.g., libpcap or Scapy) to capture packets and extract relevant information for analysis.
2. Anomaly Detection Algorithms: Develop anomaly detection algorithms to identify unusual network behavior. This can involve statistical analysis, machine learning techniques, or rule-based systems to detect deviations from normal traffic patterns, such as unexpected traffic volume, unusual protocols, or abnormal communication patterns.
3. Alert Generation: Design an alert generation mechanism that raises alerts or notifications when suspicious network activities are detected.
4. *(Optional)* The app should provide clear and actionable alerts to network administrators, indicating the type of anomaly detected and the affected systems.
5. *(Optional)* Visualization and Reporting: Create a user-friendly interface that visualizes network traffic and detected anomalies. Provide graphical representations, statistical analysis, and log reports to facilitate monitoring and analysis of network behavior.
6. *(Optional)* Real-time Monitoring: Implement real-time monitoring capabilities to continuously analyze network traffic and detect anomalies as they occur. Ensure the app can handle high network traffic volumes and provide timely alerts without significant performance impact.

Attack Simulation App

Description: Develop an attack simulation app that can connect to the network traffic anomaly detection app mentioned above. The attack simulation app will simulate popular attacks to test the effectiveness of the detection system, helping evaluate its capabilities and fine-tune its detection mechanisms.

Features:

1. Attack Simulation: Implement at least two of attack simulation techniques, “refer to the attacks’ list at the end of this document”.
2. Traffic Generation: Generate network traffic that simulates the behavior of the selected attack. The app should be able to generate various types of attack traffic, such as excessive requests, malicious payloads, or unauthorized access attempts.

3. Interaction with Detection System: Connect the attack simulation app to the network traffic anomaly detection app. The attack simulations should trigger the detection system, allowing it to analyze and detect the simulated attacks.
4. Evaluation and Analysis: Analyze the detection system's response to the simulated attacks. Assess its accuracy, sensitivity, and effectiveness in identifying and mitigating the simulated threats.
5. Reporting: Generate reports summarizing the results of the attack simulations and the detection system's performance. Include details about the detected attacks, false positives/negatives.

Choose at least two of the following popular attacks that can be generated by *"the attack simulation app"* and detected by *"the network monitoring application":*

1. Distributed Denial of Service (DDoS) Attacks: DDoS attacks involve overwhelming a target system or network with a flood of traffic from multiple sources, rendering it inaccessible. Anomaly detection algorithms can detect the sudden increase in traffic volume or abnormal traffic patterns associated with DDoS attacks.
2. Port Scanning: Port scanning is the process of systematically scanning a target system's ports to identify potential vulnerabilities. Network monitoring can detect and raise alerts for unusual port scanning activities, such as a high number of connection attempts to various ports from a single source.
3. Malware Command-and-Control (C2) Communication: Malware often communicates with command-and-control servers to receive instructions and transfer stolen data. Network monitoring can identify suspicious communication patterns, unusual domains, or known C2 server IP addresses, indicating potential malware infections.
4. SQL Injection Attacks: SQL injection attacks involve injecting malicious SQL statements into vulnerable web applications to manipulate databases or gain unauthorized access. Network monitoring can detect unexpected SQL traffic patterns or attempts to exploit known vulnerabilities in web applications.
5. Cross-Site Scripting (XSS) Attacks: XSS attacks inject malicious scripts into web pages viewed by users, potentially compromising their browsers or stealing sensitive information. Network monitoring can detect and raise alerts for unusual HTTP requests or responses containing potential XSS payloads.
6. Man-in-the-Middle (MitM) Attacks: MitM attacks intercept and manipulate communication between two parties to eavesdrop, alter data, or steal information. Network monitoring can detect anomalies in network traffic, such as unexpected changes in encryption protocols, unauthorized certificate changes, or ARP spoofing.
7. Unauthorized Access Attempts: Network monitoring can detect repeated unsuccessful login attempts, brute-force attacks, or unusual access patterns, indicating potential unauthorized access attempts to systems or applications.
8. Data Exfiltration: Network monitoring can identify large or unusual data transfers, unexpected outbound connections to suspicious IP addresses, or abnormal use of network protocols, signaling potential data exfiltration attempts.