Security Goals & Requirements

Scope

EVERSION Technologies has developed a system aimed at efficiently collecting, storing, and processing sensor data. This solution includes a mobile application that connects to sensors via Bluetooth, retrieves measured values, and transmits them to the backend for storage and further processing. Authentication services from Firebase are integrated into the mobile application to facilitate user registration and authentication before accessing its functionalities.

Furthermore, the company's public website serves as a valuable resource, providing general information about their products. The website also hosts a user-centric dashboard that displays all measurements. Firebase authentication services are implemented in the web application to ensure secure access for users. All user and measurement data are stored in the backend databases.

The scope of all security objectives and requirements outlined in this document encompasses all the aforementioned components. Specifically, it includes the sensors themselves, the mobile application, the web application, the Firebase authentication service and the Firestore database, the backend server and the backend databases (InfluxDB for storing raw sensor data, MongoDB for storing aggregated data)

Security requirements defined by the 60601-4-5 standard

Listed in separate excel file.

Additional possible security requirements

Confidentiality

Sensors

- Ensure that the data transmitted by the sensors is encrypted during communication to prevent unauthorized access to sensitive information.
- Implement measures to safeguard sensor configuration settings and firmware from unauthorized access or tampering.

Mobile Application

- Ensure that user credentials, authentication tokens, and sensitive data stored locally on the device are encrypted to prevent unauthorized access.
- Implement secure communication protocols (e.g., HTTPS) to encrypt data transmitted between the mobile application and backend servers, protecting it from interception.

Web application

- Ensure that sensitive user data, such as login credentials and personal information, is encrypted both in transit (using HTTPS) and at rest (when stored in databases).

Firebase Authentication Service and database

- Ensure that user authentication credentials (e.g., passwords, tokens) are stored securely using strong encryption algorithms.
- Implement secure hashing algorithms and salting techniques to protect user passwords from unauthorized access or manipulation.
- Implement measures to protect sensitive user data during transmission between the client application and Firebase servers, such as using HTTPS.

Backend Server and Databases

- Ensure that sensitive data stored in the backend databases (e.g., user information, sensor data) is encrypted both at rest and in transit.
- Implement access controls to restrict access to confidential data, allowing only authorized users or processes to view or modify it.

Integrity

Sensors

- Guarantee that the data collected by the sensors remains unchanged and authentic throughout the transmission process.

Mobile Application

- Employ digital signatures or message authentication codes (MACs) to verify the integrity of data received from the backend servers, ensuring that it has not been tampered with during transit.
- Implement mechanisms to detect and prevent unauthorized modifications to the application's code and resources, safeguarding its integrity.

Web application

- Employ measures such as data validation and input sanitization to prevent injection attacks (e.g., SQL injection, XSS) that could compromise the integrity of user data or the application itself.
- Implement integrity checks and checksums to detect unauthorized modifications to web application resources, such as JavaScript files or HTML templates.

Firebase Authentication Service and database

- Employ mechanisms to detect and prevent tampering or unauthorized modifications to user authentication data stored in Firebase databases.

Backend Server and Databases

- Employ data validation and integrity checks to prevent unauthorized modifications or tampering with backend data.

Availability

Sensors

- Ensure that the sensors are operational and accessible whenever needed to collect and transmit data, enabling continuous data collection.

Mobile Application

- Design the mobile application to be resilient to network disruptions and backend server outages, providing offline functionality whenever possible.
- Implement retry mechanisms and caching strategies to handle temporary network failures and ensure seamless operation under varying network conditions.

Web application

- Design the web application to be resilient to distributed denial-of-service (DDoS) attacks and other forms of traffic spikes, ensuring uninterrupted service availability for legitimate users, for example by utilizing load balancing and redundant server configurations to distribute traffic evenly.

Firebase Authentication Service and database

- Design the Firebase authentication service to be highly available and scalable, ensuring that it can handle authentication requests reliably and efficiently, even during periods of high demand.
- Utilize redundant server configurations, load balancing, and distributed architectures to minimize downtime and maximize service availability.

Backend Server and Databases

- Design the backend server infrastructure to be highly available and resilient to failures, ensuring continuous service availability for users and applications.

- Implement load balancing, redundant server configurations, and disaster recovery plans to mitigate the impact of hardware failures or infrastructure disruptions on service availability.
- Regular backups of databases should be conducted to preserve valuable information.

Authenticity

Sensors

- Implement secure authentication mechanisms to verify the identity of the sensors before allowing them to transmit data.
- Utilize digital signatures to confirm the authenticity of sensor data and prevent spoofing or impersonation attacks.

Mobile Application

- Utilize secure authentication mechanisms (e.g., OAuth, JWT) to verify the identity of users before granting access to sensitive functionalities and data within the mobile application.
- Ensure that communication with backend servers is authenticated using cryptographic protocols to verify the authenticity of server responses and prevent man-in-the-middle attacks.

Web application

- Implement secure authentication mechanisms to verify the identity of users accessing the web application.
- Utilize cryptographic protocols and digital signatures to ensure the authenticity of server responses and prevent man-in-the-middle attacks.

Firebase Authentication Service and database

- Implement strong authentication mechanisms, such as multi-factor authentication (MFA) or email verification, to verify the identity of users accessing Firebase services.

Backend Server and Databases

- Implement strong authentication mechanisms for backend services and APIs to verify the identity of clients and prevent unauthorized access.
- Utilize cryptographic protocols and digital signatures to ensure the authenticity and integrity of data transmitted from the backend server.

Accountability

Sensors

- Maintain logs of sensor activities and data transmissions to establish accountability for any unauthorized access.
- Associate each sensor with a unique identifier or authentication token to track its actions and detect any anomalies or suspicious behavior.

Mobile Application

- Log user activities and application events to maintain an audit trail of user interactions and system actions for accountability purposes.
- Associate each user session and action with a unique identifier or timestamp to facilitate forensic analysis and investigation in the event of security incidents.

Web application

- Log user activities, system events, and administrative actions to maintain an audit trail of user interactions and changes to application data.
- Associate each logged event with a unique identifier, timestamp, and user identity to facilitate forensic analysis and investigation in the event of security incidents.

Firebase Authentication Service and

- Maintain audit logs of authentication events, including successful logins, failed login attempts, and account creation activities, to establish accountability and traceability.
- Associate each authentication event with a unique identifier, timestamp, and user identity to facilitate forensic analysis and investigation in the event of security incidents.

Backend Server and Databases

- Maintain detailed audit logs of backend activities, including user interactions, data access, and system events, to establish accountability and traceability.
- Associate each logged event with a unique identifier, timestamp, and user identity to facilitate forensic analysis and investigation in the event of security incidents.

Access Control

Sensors

- Implement access control mechanisms to restrict access to sensitive sensor functionalities and configurations.
- Utilize role-based access control (RBAC) or access control lists (ACLs) to define and enforce permissions for interacting with the sensors, ensuring that only authorized users or devices can modify their settings or access collected data.

Mobile Application

- Implement role-based access control (RBAC) or permissions management mechanisms to restrict access to sensitive features and data based on user roles and privileges.
- Enforce strong password policies, multi-factor authentication (MFA), and session management controls to prevent unauthorized access to user accounts and ensure that only authenticated users can access the application's functionalities.

Web

- Implement role-based access control (RBAC) to enforce granular permissions and restrict access to sensitive features and functionalities based on user roles and privileges.
- Utilize session management controls and secure session tokens to authenticate and authorize user sessions, preventing unauthorized access to protected resources.

Firebase Authentication Service and database

- Implement access control mechanisms to restrict access to Firebase authentication resources and APIs based on user roles and permissions.
- Utilize Firebase's built-in authentication rules and policies to define granular access controls and enforce restrictions on user authentication activities.

Backend Server and Databases

- Implement robust access control mechanisms to enforce least privilege principles and restrict access to backend resources based on user roles and permissions.

Safety

Sensors

- Ensure that sensors do not pose any risk of physical harm to users or their surroundings.