

# Threat Modeling of HealthKit Applications

Fernandez Ojeda Franklin 000541971

Bui The Dat Matricule

November 16, 2025

## 1 Introduction

Over the past decade, the number of mobile health applications has grown enormously, fueled by rapid technological advancements and the widespread adoption of smartphones. This expansion has transformed the digital health landscape, making mHealth apps central to personal health monitoring and management. At the same time, the growth of these applications has made them increasingly attractive targets for malicious actors.

According to the IQVIA Institute, the global mHealth ecosystem has expanded to more than 350,000 digital health applications available on major app stores (for Human Data Science, 2021; Muoio, 2021). A similar estimate is reported by the National Institutes of Health (NIH), noting that over 350,000 mHealth apps are currently accessible to the public (of Health, 2023).

Yet this rapid expansion has also significantly broadened the attack surface. Security analyses show systemic weaknesses across the sector: one study revealed that 30 widely used mobile health applications were vulnerable to API-based attacks, exposing full medical records including protected health information (PHI) and personally identifiable information (PII) (**healthleaders2021**; **securityweek2019**). A more comprehensive industry assessment found that 71% of mHealth applications contain at least one high-severity vulnerability, while 91% fail cryptographic security tests (**zimperium2024**).

These findings highlight a critical paradox: although mobile health applications provide substantial value to users and healthcare providers, they also constitute attractive targets for attackers due to the sensitivity of the data

they process. Within this ecosystem, Apple’s HealthKit framework stands out as a widely deployed solution that aggregates personal health information, enables fine-grained data sharing, and interacts with third-party apps and cloud services.

This report focuses on performing a comprehensive threat-modeling analysis of applications leveraging HealthKit. By examining the risk landscape and identifying concrete threats, we aim to answer the following central question: Does Apple HealthKit stand out from other mobile health applications in terms of security and what specific threats does it face?

## **2 Environment and Trust Zones**

In order to effectively perform threat modeling on HealthKit-based applications, we must first define the environment in which HealthKit operates. This includes identifying actors, system components, data flows, trust boundaries, and the security guarantees associated with each part of the architecture.

### **2.1 Actors and Components**

The HealthKit environment consists of several interacting components, each playing a distinct role in the collection, storage, synchronization, and exchange of health-related information. The following subsections describe these actors in detail, with a particular focus on their security properties and their role within the overall trust model.

#### **2.1.1 User Device (iPhone / Apple Watch)**

The primary hardware platform on which HealthKit operates. The device stores the entire HealthKit database locally and enforces strict security controls through iOS Data Protection mechanisms (Apple Inc., 2023a). All health data at rest is cryptographically protected and tied to the user’s device passcode. The level of protection depends on whether the device is locked or unlocked, as access to sensitive data is transparently restricted when the device transitions to a locked state. From a threat-modeling perspective, the user device represents the core trusted environment.

### **2.1.2 HealthKit Store (local)**

The central on-device repository that stores all health and fitness metrics (Apple Inc., 2023a). This includes step counts, heart-rate samples, workout summaries, sleep analysis, menstrual-cycle data, and more. The HealthKit Store is implemented as a secure, encrypted database whose encryption keys are derived from the user’s passcode and device hardware. Even Apple cannot decrypt this data. The store also manages metadata such as timestamps, data provenance (which app generated each record), and optional cryptographic signatures that ensure data integrity.

### **2.1.3 Third-party Health Apps**

External applications that interact with HealthKit via the official APIs (Apple Inc., 2023a). These apps may request read and/or write access to specific categories of health data, but only after receiving explicit user authorization. Access control is enforced through HealthKit entitlements, meaning apps cannot interact with HealthKit unless they are granted permission at both the system and user levels. Third-party apps form an important part of the threat landscape since they represent a less-trusted zone with varying degrees of security maturity.

### **2.1.4 External Health Devices**

Devices such as heart-rate monitors, glucose meters, blood-pressure cuffs, activity sensors, and other IoT health peripherals communicating via Bluetooth Low Energy (BLE) (Apple Inc., 2021). These devices do not write data directly into HealthKit; instead, they communicate with the paired iPhone or Apple Watch, which then relays the collected measurements through an app. Because BLE communication is wireless and device security varies, this constitutes an external data-source zone with its own risks, such as spoofed or tampered sensor data.

### **2.1.5 iCloud / Cloud Infrastructure**

An optional synchronization layer enabling users to keep their health data consistent across multiple devices (Apple Inc., 2023b). If enabled, iCloud can provide end-to-end encryption (E2EE) for health data, meaning only the user’s devices hold decryption keys, and even Apple cannot access the

information. When E2EE cannot be applied—e.g., if the user has not enabled two-factor authentication—health data remains encrypted in transit and at rest but is not protected end to end. iCloud introduces a separate trust boundary since it extends data storage outside the physical device.

#### **2.1.6 Healthcare Institutions**

Clinical providers, such as hospitals and laboratories, that supply medical records through the Health app’s “Health Records” integration (Apple Inc., 2023a). The data is obtained using OAuth 2.0 for authentication and transported using TLS 1.3, ensuring strong protection during transmission. Once imported, clinical data is treated identically to other HealthKit data and stored in the same secure local database. This component represents a semi-trusted external entity with formal, regulated security guarantees.

#### **2.1.7 Medical ID**

A special subset of emergency information intended for use by first responders (Apple Inc., 2023a). Unlike standard health data, Medical ID information is accessible directly from the lock screen. To enable this, it is stored in a lower data-protection class (“No Protection”), which allows availability even when the device is locked or unpowered. From a security-architecture perspective, Medical ID represents a deliberate relaxation of confidentiality constraints to ensure life-critical accessibility.

#### **2.1.8 Temporary Journals**

A transient storage mechanism used when the device is locked and HealthKit cannot write directly to the encrypted database (Apple Inc., 2023a). During activities such as workouts, newly generated samples (e.g., heart-rate data) are stored in temporary journal files that remain accessible under the device’s locked state. Once the device is unlocked, these journal files are securely merged into the primary HealthKit Store and deleted. Temporary Journals constitute a distinct component from a security standpoint because they operate under a weaker protection class and represent a transitional data state.

### **2.1.9 Health Sharing (User to user)**

An optional feature introduced in iOS 15 that allows users to share selected health metrics with family members or caregivers (Apple Inc., 2023b). Health Sharing relies on iCloud’s end-to-end encrypted communication channels, meaning that only the sender and recipient devices hold the encryption keys. Even Apple cannot decrypt the shared information. This component represents an extension of the trusted environment beyond a single user device, forming a user-to-user trust boundary mediated by secure cloud infrastructure.

## **2.2 Data Flows and Trust Boundaries**

Understanding the movement of data within the HealthKit ecosystem is essential for identifying where trust boundaries are established and where potential threats may arise. The following paragraphs describe each data flow in detail, including its security guarantees and associated risks.

### **2.2.1 User Device $\longleftrightarrow$ HealthKit Store (Local)**

Communication between the device and the local HealthKit Store represents the core trusted data flow. All health data at rest is protected using iOS Data Protection, specifically the “Protected Unless Open” class (Apple Inc., 2023a). When the device locks, the underlying file system enforces cryptographic access restrictions, making sensitive records inaccessible after roughly ten minutes (Apple Inc., 2023a). This boundary relies entirely on the user’s passcode and hardware-based key derivation, forming the most secure segment of the HealthKit environment.

### **2.2.2 HealthKit Store $\longleftrightarrow$ Temporary Journals**

When the device is locked, HealthKit cannot write directly into the encrypted database. Instead, new samples—such as heart-rate measurements generated during workouts—are written to temporary journal files (Apple Inc., 2023a). These files operate under a weaker protection class because they must remain writable while the device is locked. Once the device is unlocked, HealthKit automatically merges the journal contents into the main encrypted store and securely deletes the temporary files. This trust boundary exposes a

transitional state where data confidentiality is still enforced but less strongly than within the fully encrypted database.

### **2.2.3 HealthKit Store $\longleftrightarrow$ iCloud**

Health data can be synchronized across devices using iCloud if the user enables this option (Apple Inc., 2023b). When two-factor authentication and a sufficiently recent OS version (iOS 12+) are in place, the synchronization channel is protected by end-to-end encryption (E2EE), meaning Apple does not hold the decryption keys (Apple Inc., 2023b). If E2EE cannot be applied, iCloud still encrypts data at rest and in transit, but Apple remains capable of decrypting it under certain conditions (Apple Inc., 2023a). This introduces a significant trust boundary, as data leaves the physical device and becomes dependent on cloud-level security properties.

### **2.2.4 HealthKit $\longleftrightarrow$ Healthcare Institutions**

Clinical records are fetched from hospitals and laboratories using standardized protocols. Authentication relies on OAuth 2.0, while data transmission uses TLS 1.3 to ensure confidentiality and integrity (Apple Inc., 2021). Once imported, clinical documents are written into the same encrypted HealthKit database as other local health data (Apple Inc., 2023a). This flow introduces a semi-trusted boundary because it depends on the security posture of external medical providers.

### **2.2.5 HealthKit $\longleftrightarrow$ Third-Party Apps**

Third-party applications may request read or write access to specific HealthKit data types. Such access is tightly controlled through system-level entitlements and user-granted permissions, creating a granular access-control boundary (Apple Inc., 2023a). Importantly, apps cannot discover whether data exists if they lack permission: HealthKit simply returns an empty result set, preventing metadata-leakage attacks (Apple Inc., 2023b). Because apps vary widely in security maturity, this boundary constitutes one of the most exposed areas in the HealthKit ecosystem.

### 2.2.6 HealthKit $\longleftrightarrow$ Medical ID

Medical ID information must remain accessible on the lock screen for emergency responders. To achieve this, it is stored using the “No Protection” class, meaning it is not tied to the device passcode and remains readable even when the phone is locked or powered off (Apple Inc., 2023a). This is a deliberate weakening of confidentiality guarantees in favor of life-critical availability and introduces a specific, well-understood trust boundary.

### 2.2.7 Health Sharing (User to User)

Starting with iOS 15, users may share selected health metrics with other individuals—such as family members or clinicians—through Health Sharing (Apple Inc., 2023b). This sharing channel is always protected with iCloud end-to-end encryption, ensuring that only the sender and recipient devices hold the keys. Apple explicitly states that it cannot decrypt or access shared data (Apple Inc., 2023b). This creates an inter-user trust boundary mediated by secure cloud-based E2EE mechanisms, effectively extending HealthKit’s protection beyond the single-device environment.

## 2.3 Security Properties and Guarantees

This subsection details the main security guarantees provided by Apple HealthKit, focusing on data protection, access control, and trust mechanisms.

### 2.3.1 Data Protection Classes

The main HealthKit database uses the “Protected Unless Open” class, while management data (e.g., app permissions, connected device names) uses “Protected Until First User Authentication” (Apple Inc., 2023a). These classes define when data is accessible relative to device lock status, ensuring sensitive information is protected when the device is locked.

### 2.3.2 Temporal Journals

Temporary journals store new health entries while the device is locked (e.g., during workouts) and are protected with the same data protection class as the main HealthKit store (Apple Inc., 2023a). Once the device is unlocked, these

journals are securely merged into the main encrypted database, ensuring data continuity without compromising security.

### **2.3.3 iCloud Encryption**

Health data synced to iCloud benefits from encryption in transit and at rest, with optional end-to-end encryption (E2EE) if the user has iOS 12+ and two-factor authentication enabled (Apple Inc., 2023b). Without E2EE, Apple can technically access the encrypted data on its servers, but transport and at-rest encryption still provide a baseline protection.

### **2.3.4 Provenance / Authenticity**

Each health record includes metadata indicating the source application, timestamps, and optionally cryptographic signatures using CMS (Cryptographic Message Syntax) (Apple Inc., 2023a). This ensures data integrity and allows detection of tampering or unauthorized modifications.

### **2.3.5 App Access Control**

Third-party apps must request HealthKit entitlements and obtain explicit user permission for each category of health data (e.g., step count, heart rate, sleep analysis, calorie intake, menstrual cycle, workout summaries) (Apple Inc., 2023a). Access is granular: if a user denies permission for a specific data type, the app will receive an empty result rather than discovering that data exists. Permissions are revocable at any time, and apps cannot infer other apps' granted permissions. This mechanism enforces a strict trust boundary between HealthKit and potentially less-trusted third-party applications.

### **2.3.6 Medical ID Accessibility**

Emergency medical information is stored with a lower protection class ("No Protection") to ensure accessibility from the lock screen even when the device is locked (Apple Inc., 2023a). This is a deliberate trade-off prioritizing life-critical availability over confidentiality.



### **2.3.7 User to user Sharing**

HealthKit allows users to share selected health data with other users via iCloud, protected by end-to-end encryption (iOS 15+, 2FA) (Apple Inc., 2023b). Only the sender and recipient possess the decryption keys, ensuring privacy even from Apple servers.

## **3 STRIDE Threat Analysis for Apple HealthKit**

This section applies the STRIDE threat modeling methodology to the Apple HealthKit ecosystem. Each threat category is analysed in relation to concrete HealthKit components and data flows, with references to Apple’s official security documentation (Apple Inc., 2021, 2023a, 2023b) and recent studies on digital health systems (for Human Data Science, 2021; Muoio, 2021; of Health, 2023). Table 1 summarizes the main threats, impacted assets, and existing mitigations.

### **3.1 Overview**

HealthKit handles one of the most sensitive categories of personal information: health and biomedical data. It integrates multiple actors (user devices, external sensors, clinical institutions, third-party applications) and crosses several trust boundaries (local database, BLE interfaces, cloud synchronization, user-to-user sharing). Such an environment makes STRIDE a relevant framework for identifying security weaknesses and residual risks.

<b>Threat Category</b>	<b>Examples in HealthKit Context</b>	<b>Existing Mitigations (Apple)</b>
<b>Spoofing</b>	Fake BLE health sensors; malicious apps impersonating legitimate ones; forged OAuth identities for Health Records.	BLE pairing with authenticated channels; app entitlements; OAuth 2.0 with secure tokens; device-bound keys.
<b>Tampering</b>	Manipulation of workout data, step counts, or imported medical records; modification of temporary journal files during device lock.	Encrypted HealthKit Store; data provenance metadata; integrity-protected records; protected journal merging on unlock.
<b>Repudiation</b>	Third-party apps denying data creation/deletion; unclear audit trail for user-to-user Health Sharing.	HealthKit stores provenance for each record; system logs; enforced app identifiers; signed clinical data.
<b>Information Disclosure</b>	Unauthorized access to health metrics; cloud interception; access by apps without permission; Medical ID exposed on lock screen.	Data Protection classes; E2EE in iCloud (opt-in); granular user permissions; “empty results” for unauthorized queries.
<b>Denial of Service</b>	BLE jamming; corrupted journal preventing DB merge; iCloud sync delays blocking data consistency; malicious apps flooding HealthKit with writes.	Process isolation; watchdog limits; safe recovery of journals; robust cloud retry mechanisms.
<b>Elevation of Privilege</b>	App writing data without proper permission; exploiting BLE device vulnerabilities; bypassing user consent screens.	Strict HealthKit entitlements; OS-level permission gating; sandboxing; App Store review enforcing health-data rules.

Table 1: STRIDE threat categories applied to Apple HealthKit.

## 3.2 Detailed STRIDE Analysis

### 3.2.1 Spoofing Identity

Spoofing concerns the impersonation of legitimate actors. In the context of HealthKit, this primarily affects BLE health peripherals, where low-cost sensors may be impersonated or replayed in order to inject falsified measurements (Apple Inc., 2021). Third-party applications also represent a risk if a malicious app attempts to pose as a trusted health app. Similarly, spoofed OAuth servers operated by malicious actors could be used to phish user credentials during the Health Records import process.

**Mitigations:** Secure BLE pairing, device-bound cryptographic identities, strict HealthKit entitlements, OAuth 2.0 with secure token endpoints, and App Store security review contribute to preventing impersonation attempts (Apple Inc., 2021, 2023a).

### 3.2.2 Tampering

Tampering refers to unauthorized modification of health data. Although the local HealthKit database is encrypted and tied to hardware keys, journal files created while the device is locked may represent a temporary target before they are merged into the main database (Apple Inc., 2023a). Third-party apps may also attempt to modify existing records without authorization, and external BLE devices could potentially transmit forged measurements.

**Mitigations:** Encryption at rest, integrity metadata for each record, verification during journal merging, and provenance tracking collectively reduce the likelihood of successful data modification (Apple Inc., 2023a).

### 3.2.3 Repudiation

Repudiation concerns disputes regarding who performed specific actions. A third-party application may deny having written or deleted a health record, while a user might dispute the origin of imported clinical lab results. Additionally, Health Sharing (user-to-user) does not provide fine-grained read receipts, which may limit legal accountability in certain contexts.

**Mitigations:** Provenance metadata linking each record to app identifiers, the use of signed and timestamped clinical documents, and system-level logging help provide verifiable evidence of actions performed within the HealthKit ecosystem (Apple Inc., 2023a, 2023b).

### 3.2.4 Information Disclosure

Information Disclosure concerns unauthorized access to sensitive health data. Key risks include the possibility of an application requesting or receiving excessive permissions, weak iCloud configurations (such as a lack of two-factor authentication) preventing end-to-end encryption, interception of BLE communications between sensors and devices, and intentional lock-screen disclosure of Medical ID information, which represents a usability–privacy tradeoff.

**Mitigations:** HealthKit relies on a granular permission model, returns empty results when an app is unauthorized, enforces end-to-end encryption when iCloud conditions are met, secures BLE channels, and applies strong Data Protection classes to prevent disclosure of sensitive information (Apple Inc., 2023a, 2023b).

### 3.2.5 Denial of Service

Denial of Service (DoS) encompasses actions that could reduce the availability of health data. Examples include BLE jamming that disrupts the collection of measurements, corruption of journal files preventing database merging after device unlock, cloud sync congestion delaying health data propagation, or malicious applications performing excessive write operations to HealthKit.

**Mitigations:** HealthKit mitigates these risks through process isolation, journal-recovery mechanisms, cloud retry algorithms, sandboxing, and OS-level watchdog mechanisms that limit abusive behavior (Apple Inc., 2021, 2023a).

### 3.2.6 Elevation of Privilege

Elevation of Privilege refers to cases where an application or attacker obtains more privileges than intended. This may involve exploiting OS vulnerabilities to bypass HealthKit permission prompts, requesting broader permissions than necessary, or abusing insecure BLE peripherals to inject privileged data types into the HealthKit store.

**Mitigations:** Mandatory entitlements, OS-enforced consent screens, strict application sandboxing, and App Store security vetting reduce the risk of privilege escalation (Apple Inc., 2021, 2023a).

## 4 Biblio

### References

- Apple Inc. (2021). Apple platform security guide (2021).  
Apple Inc. (2023a). Apple platform security.  
Apple Inc. (2023b). Health privacy: How apple protects your health data.  
for Human Data Science, I. I. (2021). *Digital health trends 2021: Innovation, evidence, regulation and adoption*. IQVIA. <https://www.iqvia.com/insights/the-iqvia-institute/reports/digital-health-trends-2021>  
Muoio, D. (2021). Digital health apps balloon to more than 350,000 available on the market, according to iqvia report. *MobiHealthNews*. <https://www.mobihealthnews.com/news/digital-health-apps-balloon-more-350000-available-market-according-iqvia-report>  
of Health, N. I. (2023). Use of mobile health applications [NIH reports more than 350,000 mHealth applications available to the public]. <https://www.ncbi.nlm.nih.gov/books/NBK595384/>