Specification plan for an **ECG-ID authentication system** using Apple Watch and companion iPhone

Authored by Derrick Hodge

Email: derrick@hodgedomain.com

Background: ECG sensor behaviour and data structure

- **Single-lead ECG measurement on Apple Watch:**
  Apple Watch Series 4 and later (and Apple Watch Ultra) include an **electrical heart sensor**. When the user places a finger on the Digital Crown, a closed circuit is formed between the heart and both arms. The back of the watch acts as a positive electrode and the Digital Crown as a negative electrode, producing a bipolar **lead-I ECG** similar to a single lead of a 12-lead ECG[1]. A typical recording lasts **30 seconds** and the ECG app classifies the rhythm as sinus, atrial fibrillation (AFib), bradycardia or tachycardia[1]. Apple's clinical trial measured **99.6 % specificity and 98.3 % sensitivity** for AFib classification compared with a 12-lead ECG[2]. The electrical sensor measures the heart signal every second and captures electrical impulses across the chest with higher fidelity when the user touches the Digital Crown[3].
- **HealthKit ECG data structure:** HealthKit stores each electrocardiogram sample with fields such as AverageHeartRate, Classification and SamplingFrequency. Each sample includes an array of sub-samples containing the MicroVolts and the TimeSinceSampleStart for each voltage measurement[4]. Developers can query raw voltage measurements using HKElectrocardiogramQuery; each measurement contains the voltage in microvolts and the time offset[5].
- **Secure processing of biometrics:** Apple devices use a **Secure Enclave**, a dedicated subsystem with its own boot ROM and AES engine, to protect sensitive keys and biometric templates[6]. The biometric security architecture separates the sensor from the Secure Enclave; the sensor captures the biometric and transmits encrypted data to the Secure Enclave, which processes and stores a template. During matching, the Secure Enclave compares the newly captured sample to the stored template and returns only a success/failure result[7].
- **Secure Triplet Loss:** The proposed system assumes that a neural network has been trained with **Secure Triplet Loss** (Pinto 2020). The method extends the traditional triplet loss by binding each feature vector with a cryptographic key (two keys are used, k_1 and k_2). Given anchor, positive and negative ECG samples, the network produces five representations; distances are computed between same-identity/same-key and mismatched key pairs. The loss encourages small distances when both identity and key match and large distances otherwise, enforcing **template cancelability** and **non-linkability**[8]. During inference, concatenating a normalized key with the feature vector before the dense layers ensures that stolen templates can be revoked by simply changing the key[9].

High-level system architecture

The ECG-ID authentication system uses the Apple Watch as a biometric capture and inference device and the iPhone as a secure storage and user-interface hub. The system is a multi-factor authentication mechanism (serving as an alternative to Face ID/Touch ID) that verifies a user's identity by comparing a **30-second single-lead ECG** to a stored template.

1. **Sensors and data acquisition**

2. **ECG capture on the Watch:** A custom watchOS app uses **HealthKit** (or **SensorKit** when available) to start an electrocardiogram session. The user is prompted to rest their forearm on a table and place one finger on the Digital Crown. The app records a 30-second single-lead ECG in lead I configuration, capturing microvolt values at the sampling frequency. The capture runs offline; no data leaves the watch until user consent is given.

3. **Signal pre-processing:** The watch app applies low-pass and notch filters (e.g., 30 Hz low-pass and 50/60 Hz notch) to remove muscle artefacts and mains interference. It normalizes the amplitude and segments the signal into fixed-length windows. R-peak detection or morphological analysis may be used to align heartbeats and reduce variation. The processed signal forms the input vector for the neural network.
4. **Key injection:** A user-specific secret key (16–32 bytes) is fetched securely from the Secure Enclave via **CryptoKit** or derived using password-based key derivation (e.g., from the user's passcode). This key is concatenated (after normalization) with the ECG features to form the neural network input, following the Secure Triplet Loss training paradigm[9].

5. **On-device inference and template generation**

6. **Core ML model:** The pre-trained ECG identification network is packaged as a **Core ML** model and deployed to the watch app. Because the model was trained with Secure Triplet Loss, it expects an ECG feature vector and a normalized key as input and outputs a fixed-length **embedding**. The embedding can be transformed to an **alphanumeric code** by encoding the vector using Base32/hexadecimal; however, the numerical vector itself is used for similarity comparison.

7. **Template production:** During **enrollment**, the user performs several ECG recordings to capture intra-person variability (e.g., three sessions at different times). Each recording is passed through the model to produce embeddings. The watch or phone aggregates these embeddings (e.g., by averaging and normalizing) to generate the **ECG template**. Only the hashed template (embedding) and the key (or key derivation salt) are stored; **raw ECG data** is discarded after processing.
8. **Cancelability and non-linkability:** If the template is compromised or the user wants to revoke it, the system simply generates a new random key and re-enrolls. Changing the key modifies the embedding space because the model binds features to the key[8], making previous templates useless. This ensures cancelability and prevents linking templates across applications or services.

9. **Secure template storage**

10. **Storage in Secure Enclave/keychain:** After enrollment, the user's ECG template is encrypted and stored in the Secure Enclave or in the iPhone's keychain with the kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly attribute. Since the Secure Enclave architecture keeps biometric templates isolated and performs matching internally[7], storing the template inside the enclave prevents extraction by the operating system or apps.

11. **Device pairing and synchronization:** When a user pairs a new watch or phone, the template can be migrated securely using Keychain data protection classes and encrypted CloudKit containers. Migration requires successful authentication via the existing factor (e.g., Face ID/Touch ID) and user consent.

12. **Authentication flow**

13. **Capture and inference:** To unlock the phone or authorize a sensitive action, the user opens their ECG-ID app or triggers the system. The watch vibrates to signal an ECG capture. The user touches the Digital Crown and holds still for **30 seconds**[1]. The watch app collects raw voltage data via HealthKit, pre-processes it, concatenates the stored key, runs inference using the Core ML model and obtains an embedding.

14. **Matching:** The watch transmits the new embedding to the Secure Enclave on the iPhone using **WatchConnectivity** and an encrypted channel. The Secure Enclave computes the Euclidean distance between the new embedding and the stored template. If the distance is below a dynamic threshold (derived from equal error rate and user-specified security level), the Secure Enclave returns a success; otherwise a failure. Only the match/no-match result is returned to the OS, preserving privacy[7].
15. **Fallback and multi-factor:** Because ECG signals may vary with stress, posture or arrhythmia, the system should support fallback to Face ID/Touch ID or passcode. A combination rule (e.g., requiring both ECG match and Face ID success for high-risk actions) can provide multi-factor security. If the watch cannot capture a valid ECG (e.g., due to poor contact), the system gracefully falls back.

16. **User interface (UI) and experience**

17. **Enrollment UI:** The iPhone companion app guides the user through the enrollment process. It explains why ECG data is needed, obtains explicit HealthKit permission and instructs the user to perform multiple ECG recordings. It informs the user that the raw ECG is not stored and that the final template is protected by the Secure Enclave.

18. **Authentication UI:** The lock screen displays a small icon prompting the user to touch the watch's crown. Progress feedback is shown on both watch and phone. After 30 seconds, the phone indicates success or failure. In case of failure, the user can retry or fall back to another factor. Settings allow users to manage keys (e.g., regenerate, disable), view when their ECG template was last used, and delete enrollment data.

19. **Integration with Apple frameworks**

20. **HealthKit/SensorKit:** Use HKElectrocardiogram to query existing ECG recordings or HKElectrocardiogramBuilder (introduced in watchOS 10) to collect real-time ECG samples. For research apps requiring continuous streaming, **SensorKit** provides SRElectrocardiogramSensor, but this requires entitlements and a declared usage reason in the Info.plist. The Info.plist must include com.apple.developer.sensorkit.reader.allow and the NSHealthUpdateUsageDescription and NSHealthShareUsageDescription keys to request permission.

21. **Core ML and Accelerate:** The on-watch model uses Core ML for inference. Pre-processing filters can be implemented using the Accelerate framework for performance.
22. **CryptoKit and LocalAuthentication:** Generate keys, derive symmetric keys from passcodes and interface with the Secure Enclave. Use LocalAuthentication to combine ECG-ID with Face ID/Touch ID or passcode policies.

23. **WatchConnectivity:** Securely transmit embeddings and match results between the watch and phone.
24. **Keychain & CloudKit:** Store templates and keys securely and synchronize them across the user's devices.
25. **Security and privacy considerations**
26. **Consent and transparency:** The app must clearly explain why ECG data is collected and how it will be used. As HealthKit data is sensitive, the user must grant explicit permission, and the app must abide by Apple's guidelines not to use HealthKit data for advertising or data brokerage.
27. **Data minimization:** Only the derived embedding and a salted hash are stored. Raw ECG recordings are deleted immediately after processing.
28. **Encryption in transit and at rest:** All communications between watch and phone occur over encrypted sessions. The embedding and keys are encrypted using AES‑GCM inside the Secure Enclave.
29. **Revocation and template update:** Because the Secure Triplet Loss binds the embedding to a secret key, users can revoke or update their ECG template by regenerating the key and re‑enrolling. This helps defend against template compromise and replay attacks.
30. **Spoofing and liveness detection:** The system should implement liveness detection by verifying that the ECG waveform has the expected morphology and by checking that the Apple Watch is being worn. The watch's photoplethysmography (PPG) sensor and accelerometer can be used to detect if the watch is on the wrist and if the user is stationary.
31. **Compliance:** If used for medical or regulated applications, ensure compliance with HIPAA/GDPR. Document data handling, retention and deletion policies.
32. **Future enhancements and research opportunities**
33. **Multiple leads and improved accuracy:** Although the Apple Watch normally records a single lead[1], placing the watch on different body locations and touching the Digital Crown with various fingers can record leads II/III[10]. Collecting multi‑lead ECGs could improve identification accuracy.
34. **Continuous authentication:** Instead of only authenticating at unlock time, the system could monitor the user's ECG continuously (with user consent) using the PPG sensor and periodic ECG snapshots to maintain a dynamic confidence score.
35. **Federated learning:** Future versions could update the neural network on the device using federated learning, keeping raw ECGs on the watch and sending only gradient updates to a central server.
36. **Integration with passkeys:** The ECG‑ID can be integrated into **passkey** frameworks to authenticate to web services. Once the watch verifies the user, it can sign a FIDO2 assertion with the user's private key stored in the Secure Enclave.

Conclusion

Leveraging the Apple Watch's electrical heart sensor and the Secure Triplet Loss-trained neural network, the proposed ECG-ID system enables robust biometric authentication that is cancelable and privacy-preserving. The system uses watchOS and iOS frameworks such as HealthKit, Core ML, CryptoKit, LocalAuthentication and WatchConnectivity to capture ECG signals, generate secure embeddings, store them in the Secure Enclave and match them during authentication. Compliance with Apple's biometric security architecture[7] and careful user-experience design ensure that ECG-ID can be a viable complement to existing Face ID/Touch ID methods.

References

[1] [10] Using the Apple Watch to Record Multiple-Lead Electrocardiograms in Detecting Myocardial Infarction: Where Are We Now? - PMC

https://pmc.ncbi.nlm.nih.gov/articles/PMC9427050/

[2] Take an ECG with the ECG app on Apple Watch - Apple Support

https://support.apple.com/en-us/120278

[3] Monitor your heart rate with Apple Watch - Apple Support

https://support.apple.com/en-us/120277

[4] Apple HealthKitV2 Electrocardiogram Export Format – MyDataHelps™ Designer Help Center

https://support.mydatahelps.org/hc/en-us/articles/4412383294099-Apple-HealthKitV2-Electrocardiogram-Export-Format

[5] ios - How to store ECG data on apple healthkit? - Stack Overflow

https://stackoverflow.com/questions/53376307/how-to-store-ecg-data-on-apple-healthkit

[6] Secure Enclave - Apple Support

https://support.apple.com/guide/security/secure-enclave-sec59b0b31ff/web

[7] Biometric security - Apple Support

https://support.apple.com/guide/security/biometric-security-sec067eb0c9e/web

[8] [9] jpinto2020iwbf.pdf

https://jtrpinto.github.io/files/pdf/jpinto2020iwbf.pdf