



Quorum Seal: Cross-Sensor Challenge and Response Attestation for Compromise Detection with Adaptive Multi-Surface Verification

Saravanakumar M¹, Habib Dhulfikhar H²

¹ Professor, Department of Computer Science and Engineering, Vandayar Engineering College, Thanjavur, Tamil Nadu, India.

² Student, Department of Computer Science and Engineering, Vandayar Engineering College, Thanjavur, Tamil Nadu, India.

To Cite this Article: Saravanakumar M¹, Habib Dhulfikhar H², “Quorum Seal: Cross-Sensor Challenge and Response Attestation for Compromise Detection with Adaptive Multi-Surface Verification”, *Indian Journal of Computer Science and Technology*, Volume 05, Issue 01 (January-April 2026), PP: 239-242.



Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the [Creative Commons Attribution License](https://creativecommons.org/licenses/by-nc-nd/4.0/); Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: Smartphone compromise detection is challenging because a single software-only check can be hidden, replayed, or forged by a capable attacker. This paper presents Quorum Seal, an evidence-based mobile trust attestation framework that evaluates whether a device is trustworthy enough for sensitive actions such as login, payment approval, marks entry, and protected data access. Quorum Seal uses a nonce-based challenge-response protocol, on-device sensor capture, compact statistical feature extraction, and server-side weighted quorum verification to classify a session as Trusted, Suspicious, or Untrusted. The complete system adds cross-sensor conflict fingerprinting, Adaptive Challenge Escalation, dynamic quorum adaptation for missing or low-quality sensors, and an entropy analyzer to detect low-variance or synthetic motion patterns. Each verification produces an explainable evidence record retrievable via `/evidence/{id}`, exposing the checks, penalties, and reasons behind the verdict. The prototype is implemented with a Flutter Android client and a Fast API backend exposing `/challenge`, `/verify`, and `/evidence/{id}`, and has been validated using reproducible evidence outputs, including real-device runs. Rather than claiming absolute malware diagnosis, Quorum Seal provides a practical and auditable transaction-time trust decision that raises attacker cost and supports safer access-control policies.

Key Words: Mobile attestation; Challenge-response; Sensors; Device trust; Quorum scoring; Adaptive verification.

I. INTRODUCTION

Modern mobile devices routinely authorize sensitive operations, yet many deployed defenses still rely on a small number of static indicators such as root flags, app signatures, or environment checks. These indicators are useful but insufficient when applications are repackaged, instrumentation frameworks are injected, or sensor responses are replayed. Recent platform services improve confidence in app and device state, but practical risk assessment still benefits from additional context and evidence collected during a live transaction.

Quorum Seal addresses this gap through a multi-proof attestation design. The central idea is simple: a single signal may be spoofed, but forging multiple independent signals consistently under a fresh challenge is substantially harder. Quorum Seal therefore combines challenge freshness, sensor-driven livens evidence, feature-level verification, and weighted quorum scoring to produce an explainable trust verdict. The framework is not positioned as an antivirus that removes malware from the phone; instead, it functions as a trust gate that determines whether the device should be allowed, challenged further, or blocked for a sensitive action.

The contribution of this work is twofold. First, it presents an implementable Core pipeline consisting of challenge generation, sensor capture and feature extraction, and quorum verification with a decision engine. Second, it defines an Enhanced modules extension that adds conflict fingerprinting across sensors, adaptive challenge escalation, dynamic quorum adaptation for missing or weak sensors, entropy-based anti-spoof checks, and an evidence panel that exposes the reasons behind the verdict.

II. MATERIAL AND METHODS

This work is a systems design and prototype implementation study rather than a clinical or population-based trial. The prototype was developed as a client-server attestation framework. The mobile client was implemented in Flutter on Android, while the backend verification service was implemented using Fast API in Python. At the time of writing, the locally tested backend exposes two operational endpoints, namely `/challenge` and `/verify`. The complete system includes on top of the same architecture, including evidence storage, conflict fingerprinting, adaptive challenge escalation, dynamic quorum adaptation, and entropy-based anti-spoof checks.

Study Design: Design-oriented prototype study with modular implementation and server-assisted attestation.

Study Location: Development and local execution environment using an Android handset or emulator for client-side integration

and a Fast API server for attestation and verification flows.

Study Duration: Implementation and validation of the complete Quorum Seal prototype with adaptive verification modules and reproducible evidence outputs.

System Architecture

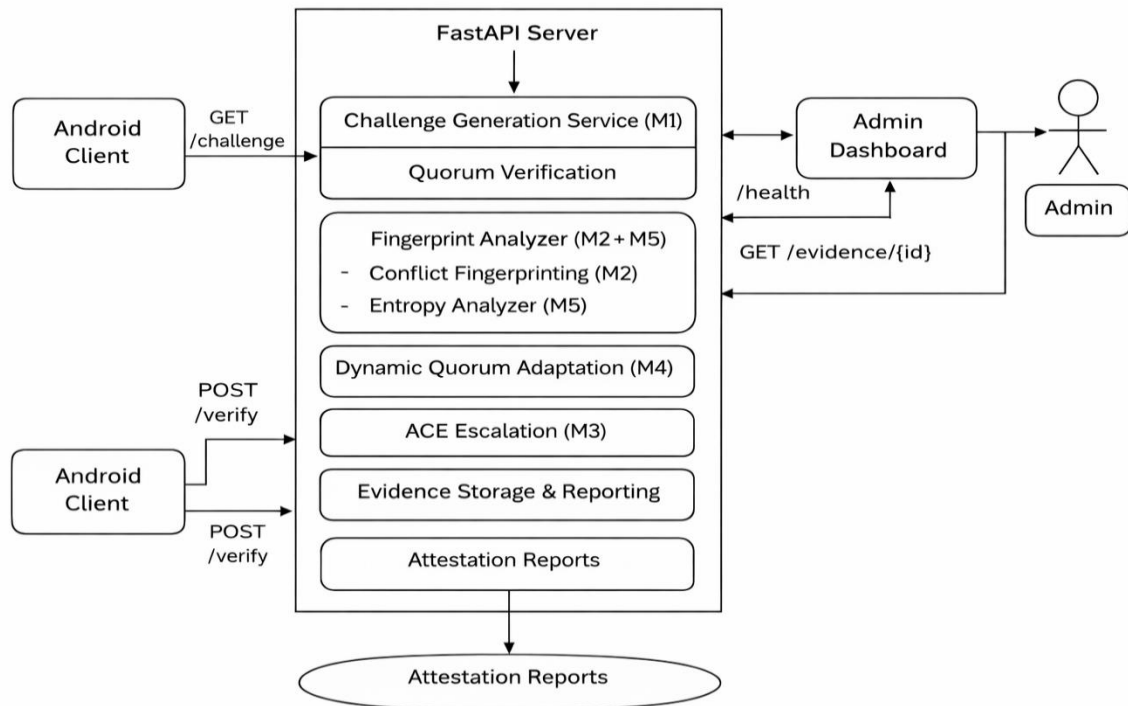


Figure 1: Quorum Seal complete system architecture: server issues challenges (/challenge), client uploads proofs (/verify), evidence is stored and retrieved (/evidence/{id}), with conflict fingerprinting, entropy analysis, adaptive escalation, and dynamic quorum adaptation.

System Components: The framework contains three core pipeline modules: (i) Challenge Generation, (ii) Sensor Capture and Feature Extraction, and (iii) Quorum Verification and Decision Engine. Enhanced modules add five enhancement modules: cross-sensor conflict fingerprinting, Adaptive Challenge Escalation, explainable evidence logging, dynamic quorum adaptation, and sensor entropy analysis.

Module	Role in QuorumSeal
Challenge Generation	Creates a fresh nonce, challenge identifier, timing constraints, and requested sensing recipe.
Sensor Capture and Feature Extraction	Collects sensor readings and converts them into compact statistical features
Quorum Verification and Decision Engine	Validates freshness, applies quality gates, and computes module scores and trust score Trusted/Suspicious/Untrusted.
Explainable Evidence Panel	Stores session evidence and exposes reasons, checks, penalties, and escalation history.
Cross-Sensor Conflict Fingerprinting	Detects inconsistent relationships between sensor streams indicating spoof/replay
Adaptive Challenge Escalation	Requests stronger challenge levels when evidence is borderline or conflicting.
Dynamic Quorum Adaptation	Adapts quorum/weights when sensors are missing or low quality
Sensor Entropy Analyzer	Flags low-variance/flat line or overly regular sensor patterns.

Table 1: Functional Modules of Quorum Seal

III. PROCEDURE METHODOLOGY

Quorum Seal begins when the backend issues a fresh challenge containing a nonce, a time limit, and the required sensing recipe. The mobile client receives the challenge, triggers the requested sensing flow, captures readings from available sensors, and computes compact statistical features (e.g., mean, variance/standard deviation, range, peak values, and quality indicators). The client then uploads the proof package to the backend along with the challenge identifier and metadata.

On receipt, the verification service validates nonce freshness, expiry, request consistency, and feature completeness. It applies sensor quality gates and computes module scores from multiple independent proofs. The scores are aggregated using a weighted quorum rule to produce a trust verdict for the requested action: Trusted, Suspicious, or Untrusted.

The same end-to-end pipeline also provides adaptive verification and explainability. Each session is logged as an evidence record and can be retrieved through `/evidence/{id}`. Cross-sensor conflict fingerprinting checks whether sensor relationships are physically plausible, and an entropy analyzer flags overly smooth or flatline patterns indicative of synthetic/emulated inputs. When evidence is borderline, Adaptive Challenge Escalation requests a stronger challenge level and the client reruns capture once. Dynamic quorum adaptation adjusts the active quorum and weights when some sensors are missing or low quality, ensuring the decision remains fair and robust.

IV. STATISTICAL ANALYSIS

Because this manuscript reports a design and prototype implementation, analysis is based on algorithmic scoring rather than hypothesis testing on a clinical cohort. Quorum Seal converts sensor streams into statistical features and evaluates each proof channel using acceptance rules and quality gates. A final trust score is computed as a weighted aggregation of module scores in the active quorum:

$$\text{Trust Score} = \sum (w_i \times S_i)$$

Where S_i is the module score and w_i is its weight under the current quorum configuration. Additional derived scores are recorded for explain ability and tuning, including conflict score, entropy score, and the escalation state (challenge level). These values are used for descriptive reporting, threshold tuning, and ablation-style comparison of which proofs contributed to the final verdict.

V. RESULT

Scenario	Evidence ID	Decision	Trust Score	Risk	Key reason / signal
Trusted on physical Android (release build)	7ef4ba9b-a852-4e0f-834d-0b62523ee8bd	TRUSTED	82	LOW	Real device run; expiry/nonce/quorum pass; conflict=0; entropy=0; warnings: developer_mode_enabled, app_finger_print_drift
Untrusted on physical Android (debug + flatline)	fd306d03-538a-471c-a935-7ba19c43ecac	FLAGGED	42	HIGH	Real device run; entropy flatline (accel/gyro) + debug/developer penalties → trust below threshold.
Trusted baseline (clean evidence)	b0c98798-6269-476b-a57e-d59fd82a850a	TRUSTED	100	LOW	Clean evidence
Untrusted hard-fail (expired + nonce mismatch)	0a9ee872-5e80-4051-b859-59b772d127c5	FLAGGED	0	HIGH	challenge_expired + nonce_mismatch
Suspicious: conflict fingerprint (motion mismatch)	08991465-c42c-4ed9-8662-9e792f302603	FLAGGED	65	MED	conflict: motion_intensity_mismatch (score 40)
Suspicious: escalation triggered (Level-1)	0f58c9b6-a7df-46b4-a8db-99d5056cc90b	FLAGGED	65	MED	escalate=true → next_level=2
Untrusted: strict verification result (Level-2)	f6c263b2-b6dd-4c48-b5e2-90eda7ab1de0	FLAGGED	0	HIGH	Level-2 gates + quorum_t=4
Trusted with missing sensor (dynamic quorum)	03bd6f4b-e433-443a-beb3-7716ee335525	TRUSTED	90	LOW	missing location; quorum adapted
Untrusted: quorum unavailable (insufficient sensors)	92b36110-3eeb-4d9d-9c74-b160c82d7dbe	FLAGGED	0	HIGH	dynamic_quorum_unavailable (only 1 good sensor)
Trusted: entropy pass (natural motion)	6317cd75-4f79-4bcc-bfc4-c5c735aead79	TRUSTED	100	LOW	entropy pass (natural)
Suspicious: entropy flatline (penalty + escalation)	03190049-849d-4707-80da-98ea96561220	FLAGGED	60	MED	entropy fail: flatline accel/gyro; penalty -40; ACE triggered

Table 2: Reproducible Quorum Seal evidence outputs (Core pipeline + Enhanced modules M1–M5).

The implemented Quorum Seal prototype successfully demonstrates the intended client–server attestation loop in a local environment. The Flutter Android client and Fast API backend support challenge issuance, verification, and evidence retrieval through the operational /challenge, /verify, and /evidence/{id} endpoints. This confirms the feasibility of the proposed architecture, including server-driven challenge generation, sensor-driven evidence collection, compact feature transfer, and explainable decisioning on the backend.

At this stage, the most important outcome is architectural and functional validation rather than benchmark superiority. The prototype establishes that Quorum Seal can provide an evidence-based trust verdict without depending on a single binary indicator. By combining freshness checks, sensor quality gates, weighted quorum scoring, and explainable evidence logging, the system produces practical trust outcomes that can be used to allow, step-up, or block sensitive actions.

The manuscript avoids fabricating benchmark claims; instead, it reports reproducible evidence records captured from the running system. The validated outcome is that the end-to-end pipeline and its adaptive modules—conflict fingerprinting, entropy analysis, dynamic quorum adaptation, and adaptive challenge escalation—operate as intended and generate explainable evidence for Trusted, Suspicious, and Untrusted sessions.

VI. DISCUSSION

Quorum Seal should be interpreted as a trust-attestation framework, not as a standalone malware cleaner. Its value lies in reducing the risk of sensitive operations by making them conditional on the quality of device evidence available at the moment of use. This distinction is important in practice and in scholarly positioning: Quorum Seal does not promise perfect compromise diagnosis under every attacker model, especially if a highly privileged adversary controls the operating system or can instrument the app deeply. Instead, it raises the cost of deception by forcing the adversary to forge several independent proofs consistently.

The use of sensors is justified by the need for liveness and diversity of evidence. Software-only indicators are valuable but can sometimes be masked or replayed. By contrast, fresh sensor-driven challenge-response data introduces a physical component that is harder to reproduce under time constraints, particularly when multiple sensor relationships must remain coherent. The conflict fingerprinting and entropy modules strengthen this argument by explicitly targeting implausible cross-sensor combinations and overly regular streams typical of synthetic or emulated inputs.

Another strength of the framework is explain ability. Many security verdicts fail to support operators because they provide only a binary answer. The proposed evidence panel addresses this by exposing which module contributed to the decision, why escalation was triggered, and which evidence appeared weak. This makes the system more suitable for audits, access-control policies, and academic demonstration. Future evaluation should therefore compare not only security outcomes but also transparency, operator trust, and decision usability.

VII. CONCLUSION

Quorum Seal presents a practical path toward evidence-based mobile trust assessment for sensitive actions such as login approval, payments, and protected data access. The implemented prototype demonstrates an end-to-end challenge–response attestation pipeline in which the server issues time-bound challenges, the mobile client captures multi-sensor evidence and extracts compact statistical features, and the verifier computes an explainable trust verdict using freshness checks, quality gates, and weighted quorum scoring.

The complete system strengthens robustness through cross-sensor conflict fingerprinting, sensor entropy analysis, dynamic quorum adaptation for missing or low-quality sensors, and adaptive challenge escalation when evidence is borderline. Rather than claiming absolute malware diagnosis, Quorum Seal provides a defensible and auditable trust decision supported by reproducible evidence records retrievable through /evidence/{id}. This modular design increases attacker cost, improves transparency, and enables safer access-control decisions in real deployments. Future work will extend the client-side acquisition layer to iOS using platform-specific integrity and sensor APIs while preserving the same server-side decision engine and evidence model.

References

1. Google. Play Integrity API overview. Android Developers. Available from: <https://developer.android.com/google/play/integrity/overview>
2. Google. Returned integrity verdict format. Android Developers. Available from: <https://developer.android.com/google/play/integrity/verdicts>
3. Google. Make a standard API request. Android Developers. Available from: <https://developer.android.com/google/play/integrity/standard>
4. Apple. Device Check and App Attest documentation. Apple Developer Documentation. Available from: <https://developer.apple.com/documentation/devicecheck>
5. Apple. Establishing your app's integrity. Apple Developer Documentation. Available from: <https://developer.apple.com/documentation/devicecheck/establishing-your-app-s-integrity>
6. Apple. Security overview: Device Check and the App Attest API. Apple Developer. Available from: <https://developer.apple.com/security/>
7. Abuhamad M, Abusnaina A, Nyang D, Mohaisen D. Sensor-Based Continuous Authentication of Smartphones' Users Using Behavioral Biometrics: A Contemporary Survey. *IEEE Internet of Things Journal*. 2021; 8(1):65-84.
8. Ankergard SFJJ, et al. State-of-the-Art Software-Based Remote Attestation. *Sensors*. 2021; 21(4):1418.
9. Ehatisham-ul-Haq M, et al. Authentication of Smartphone Users Based on Activity Recognition and Mobile Sensing. *Sensors*. 2017; 17(9):2043.
10. Shuwandy ML, et al. Sensor-based authentication in smartphone: A systematic literature review. *Smart Health*. 2025; 35:100515.