Asia Pacific Academy of Science Pte. Ltd.

Metaverse 2026, 7(4), m260050.
https://doi.org/10.54517/m260050

Article

An Explainable and Verifiable Digital Twin of a Two-Site Cyber Range for Metaverse Infrastructure Assurance: Software Artifact and Field Protocol

Oleksandr Korchenko[1], Dmytro Prokopovych-Tkachenko[1,*], Kyrylo Davydenko[2], Mukafat Askerov[1], Anton Herasymenko[1], Yurii Zhurov[3]

[1] State University of Information and Communication Technologies, Kyiv, Ukraine.
[2] National Technical University "Dnipro Polytechnic", Dnipro, Ukraine.
[3] SMB Cybersecurity Advisors, United States of America.

[*]**Corresponding author:** Dmytro Prokopovych-Tkachenko, omega2417@umsf.dp.ua

CITATION

Korchenko O, Prokopovych-Tkachenko D, Davydenko K, et al. An Explainable and Verifiable Digital Twin of a Two-Site Cyber Range for Metaverse Infrastructure Assurance: Software Artifact and Field Protocol. Metaverse. 2026; 7(4): m260050. https://doi.org/10.54517/m260050

COPYRIGHT

Copyright © 2026 by author(s). *Metaverse* is published by Asia Pacific Academy of Science Pte. Ltd. This work is licensed under the Creative Commons Attribution (CC BY) license. https://creativecommons.org/licenses/by/4.0/

Abstract: Metaverse-oriented infrastructure assurance requires a virtual representation to distinguish operational degradation from manipulated state updates while retaining evidence for review. This article presents MetaTwin-Guard, an assurance backend and prospective field protocol for a documented two-site university cyber range. The confirmed environment comprises seven nominal wide-area links, an inter-site VPN, 54 wireless access points, three backup-power stations, and 25 Kali Linux workstations. The reference software processes exported or synthetic telemetry only; it performs Ed25519 event verification, replay and hash-chain checks, topology-aware state updates, a transparent median/median-absolute-deviation anomaly score, static operational rules, and a four-validator, three-of-four quorum-signed evidence chain. That chain is explicitly a single-process emulation rather than a distributed blockchain. Automated tests exercised clean-state processing, replay rejection, telemetry-tamper rejection, evidence-block tamper detection, and ledger verification. A deterministic dry run used seed 20260823, six scenarios, eight repetitions per scenario, and 17,280 generated events. Hybrid mean F1 was 0.879 for replay, where feature-only rules and robust scoring both yielded zero; scenario-specific results also exposed a weak robust-score response to a binary power transition. A derived ablation attributes the replay gain entirely to the integrity layer. These values verify designed code paths only; all conclusions are limited to software-level verification and do not constitute cyber-range field evidence. The accompanying randomized, independently reset field protocol specifies authorization, isolation, timing, metrics, stop criteria, data schemas, and statistical analysis. No physical field-run series or human-participant study has been completed; field performance remains to be established.

Keywords: Digital twin; Cyber range; Cyber-resilience; Signed telemetry; Explainable anomaly detection; Evidence provenance; Replay attack; Ed25519.

1. Introduction

Digital twins connect observations of a physical or operational system to a computational representation that can be inspected, simulated, and updated over time. Current terminology distinguishes the represented entity, its digital representation, and the information exchange that keeps the two aligned [1,2]. The value of a twin therefore depends on more than visual fidelity: the timeliness, authenticity, completeness, and semantic consistency of each state update determine whether the virtual state is fit for operational interpretation. Digital-twin testing literature likewise treats fidelity and validation as unresolved engineering concerns rather than properties conferred by the label alone [3].

A metaverse-facing twin extends this problem. Remote users, visualization

services, and autonomous components may share and act upon one representation, while the rendering layer can make a stale or manipulated state appear authoritative. Surveys of metaverse security identify identity, privacy, data integrity, availability, and cross-layer trust as central concerns [4]. In an infrastructure-assurance setting, a visually convincing state is not necessarily a trustworthy state. Delayed, replayed, modified, reordered, or falsely attributed telemetry may cause the twin to diverge from the observed network and may mislead both human operators and automated decision support.

The relevance to the metaverse is therefore located at the trust layer beneath the immersive interface rather than in the interface itself. When several remote users, visualization services, or automated agents act on one shared virtual representation, an erroneous state is not a local display defect but a shared premise for decisions. A metaverse-facing infrastructure twin consequently needs a contract that states, for every rendered element, whether it was admitted, how fresh it is, why it was flagged, and which evidence record supports it. The present work specifies and verifies the backend side of that contract. It does not demonstrate an operational immersive interface or a complete deployed digital twin, and the metaverse-facing view of the architecture described in Section 3.4 is explicitly marked as proposed.

The present work therefore uses three explicit evidence levels. First, the documented two-site cyber-range topology is treated as confirmed context. Second, an executable reference artifact is tested with unit tests and a deterministic synthetic generator. Third, a prospective field protocol specifies how the artifact can be evaluated safely on the cyber range after administrative, technical, privacy, and measurement gates have been satisfied. Synthetic verification can establish that code follows the specified logic. It cannot establish real wireless performance, provider failover behavior, production-scale throughput, distributed-ledger confirmation time, or operator usability.

The study addresses four research questions:

RQ1: How do static operational rules, a robust explainable anomaly detector, and their integrity-aware hybrid combination behave across six deterministic synthetic scenarios representing infrastructure disturbances and telemetry attacks?

RQ2: Can a cryptographic integrity gateway prevent structurally invalid, modified, out-of-sequence, or replayed events from changing the twin state while preserving an auditable decision record?

RQ3: What synchronization and evidence-append timing is observed in the single-process synthetic benchmark, and what conclusions must not be drawn from those timings?

RQ4: How should the artifact be evaluated in a future field experiment on a two-site cyber range without exposing operational services or overstating the available infrastructure?

The principal contributions are: (i) an assurance pipeline separating event admissibility, anomaly scoring, explanation, state transition, and evidence recording; (ii) a computationally transparent median/median-absolute-deviation detector and explicit static rules; (iii) Ed25519 event authentication and a quorum-signed, hash-chained evidence layer; (iv) deterministic verification across six scenarios and 48 benchmark runs; and (v) a reproducible, safety-gated field protocol grounded in the confirmed topology. The software results are reported as artifact-level evidence only. No field-performance result is claimed.

The individual mechanisms used here, namely Ed25519 signatures, SHA-256 hash chaining, median/MAD scoring, and quorum signatures, are established, and no novelty is claimed for any of them in isolation. The contribution lies in how they are composed and constrained. First, admission is formulated as a state

invariant (Equation (2)): no event can modify the twin unless it is authentic, fresh, and chain-consistent, so integrity failures are excluded from the represented state rather than merely flagged alongside it. Second, each evidence record binds the integrity outcome, the anomaly score, its feature-level explanation, and the resulting twin-state hash in one verifiable object, so that what the twin displayed can later be reconciled with why it displayed it. Third, the integrity-aware hybrid makes a class of manipulation that is invisible in feature space (replay) explicit, and the derived ablation in Section 4.2 attributes that gain to the responsible layer. Fourth, the evidence-level distinction between confirmed, implemented, verified, and planned elements is applied throughout, so that software verification is never reported as field performance.

2. Literature review

The attack surface extends beyond conventional endpoint compromise. An adversary may alter a measurement, resubmit a previously valid event, violate event order, impersonate a telemetry producer, suppress an adverse observation, or create inconsistencies between network status and the state presented by the twin. Availability faults can produce similar symptoms without malicious intent. A failed wide-area link, a degraded virtual private network (VPN), an authentication surge, or a transition to backup power can alter several telemetry dimensions at once. Cybersecurity and resilience frameworks therefore emphasize governed risk, anticipation, detection, response, recovery, and adaptation rather than a single classifier outcome [5–7].

Artificial-intelligence-based anomaly detection may reveal multivariate changes that fixed thresholds miss, but an unexplained score is difficult to use in an operational environment. Explanations should identify which observations contributed to a decision, and evaluation should test generalization and robustness rather than only in-sample accuracy [8–10]. Deterministic integrity controls remain necessary for cases that statistical inference cannot reliably detect. A replayed event can contain entirely ordinary feature values; its defect is freshness and ordering, not necessarily its feature vector.

Cryptographic signatures, canonical payload hashes, monotone sequence controls, and per-source hash links provide a complementary verification layer. Ed25519 supplies a standardized digital-signature mechanism [11], while SHA-256 provides the hash primitive used by this reference artifact [12]. These mechanisms do not replace network segmentation, key management, or operational investigation. They answer a narrower set of questions: whether a known source signed the submitted envelope, whether its canonical payload changed, whether the identifier or sequence had already been accepted, and whether the event continues the accepted source chain.

Cyber ranges are appropriate environments for investigating this class of assurance mechanism because they support realistic but controlled training and testing [13,14]. Existing work has connected digital twins with security-operations training [15], 5G cyber ranges [16], industrial Internet-of-Things environments [17], reconfigurable critical-infrastructure training [18], multi-range orchestration [19,20], and network-twin deployment methodology [21]. These studies motivate an integrated assurance backend, but they do not eliminate the need to separate implemented functions from proposed instrumentation or to distinguish deterministic software verification from physical field measurement.

A closer comparison clarifies the gap. Digital-twin cyber ranges for security-operations training [15], 5G testing [16], industrial Internet-of-Things environments [17], and critical-infrastructure education [18] primarily use the twin as an emulation and training environment; their emphasis lies on realistic scenarios and orchestration rather than on whether each state update entering

the twin is authentic, fresh, and attributable. Multi-range and orchestration studies [19,20] address the scale and composition of ranges, and network-twin methodology [21] addresses modeling and deployment of the twin itself; per-event admission control is not their primary concern. The work closest in intent combines digital twins, smart contracts, and explainable detection in a zero-touch network security framework [22]. It demonstrates the value of integrating these mechanisms, but it relies on learned detection with post hoc explanation, and its emphasis is on secure data sharing and detection rather than on gating twin-state updates by per-event admission. The present study is narrower in scope and smaller in scale than these systems and contributes no new training environment, orchestration method, or learned detector. Its distinct contribution is an explicit admission-before-state invariant, an explanation that is the exact decomposition of the score used for the decision, and an evidence record that binds both to the twin-state hash, together with a strict separation between verified software behavior and prospective field evidence.

3. Methodology

3.1. Study design and evidence boundaries

The work follows a staged technical-validation design. Evidence classes were fixed before interpretation so that architectural assumptions could not be presented as deployed capabilities.

Level A contains confirmed physical context from the supplied laboratory description: the two geographically separated sites, their boundary routers, nominal Internet connections, UniFi management controllers, access points, backup-power stations, inter-site VPN, and Kali Linux classroom. Level B contains deterministic software verification produced by the executable artifact under a fixed seed. The generator creates baseline and evaluation events with known labels, allowing the integrity gateway, detectors, twin-update logic, explanations, and evidence chain to be tested repeatably. Level C contains the planned field method and no completed measurement.

This separation governs the verbs used in the manuscript. “Confirmed” refers to documented physical components. “Implemented” refers to executable code in the accompanying package. “Verified” refers to a specified code path that passed the deterministic benchmark or automated test. “Planned” refers to a procedure that remains contingent on authorization, inventory, isolation, and measurement readiness. The term “field result” is reserved for observations acquired from the physical cyber range; none were available for this study.

3.2. Confirmed cyber-range context

The study context is a distributed cyber range operated by the University of Customs and Finance in Dnipro, Ukraine. The sites are designated Site A and Site B. The sites are geographically separated and connected through a protected site-to-site VPN. The available description supports a network-training and infrastructure-assurance study; it does not establish that the cyber range itself is a regulated critical-infrastructure facility or an operational industrial-control system.

Site A uses a Keenetic Titan boundary router and has five documented provider connections: three nominal 1 Gbit/s links and two nominal 100 Mbit/s links. Its wireless environment is administered through a UniFi CloudKey Gen2 management controller and contains 48 access points. Twelve access points have confirmed gigabit wired uplinks. The uplink characteristics of the remaining 36 access points were not supplied. Three distributed EcoFlow stations provide backup power to selected network nodes, but models, capacities, connected loads, protective limits, switching behavior, and measured autonomy were not available.

Site B uses a Keenetic Viva boundary router and has two nominal 1

Gbit/s provider connections. Its wireless environment is administered through a UniFi CloudKey Gen1 controller and contains six access points with 100 Mbit/s wired connections. Site B also includes a classroom with 25 Kali Linux workstations. Those machines are the confirmed general-purpose platform available for controlled traffic generation, local test services, artifact execution, and collection of authorized endpoint observations.

The documented environment therefore contains seven provider connections and 54 access points in total, an inter-site VPN, backup-power capability at Site A, and 25 Kali Linux workstations at Site B. Nominal provider rates are not treated as measured end-to-end throughput. A CloudKey is treated as a wireless-management-plane component, not a transit router. Router firmware, controller versions, access-point models, switches, VLANs, access-control lists, IP plan, VPN protocol and maximum transmission unit, and Kali workstation hardware remain unresolved. A general-purpose compute node at Site A has not been confirmed. These limitations constrain the field design. **Figure 1** summarizes the confirmed topology and proposed overlay, while **Table 1** separates documented components from unresolved or proposed elements.

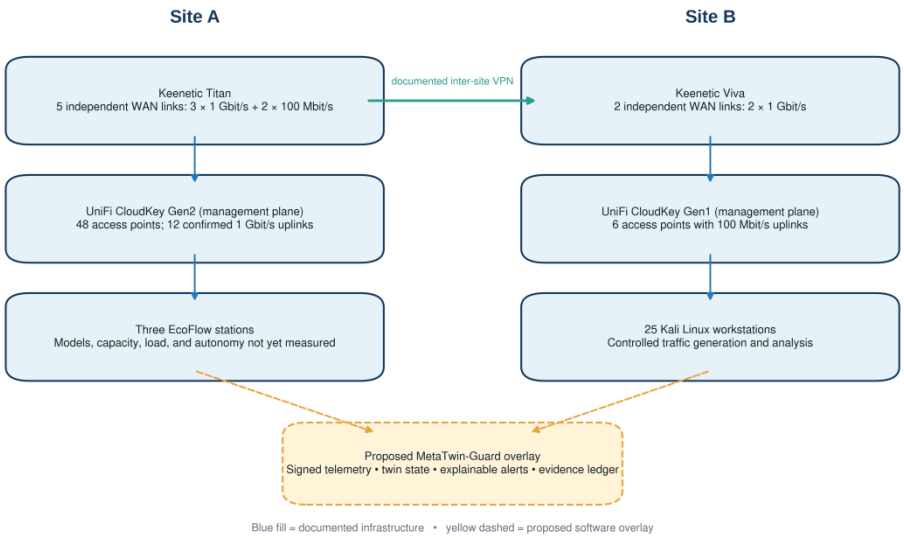


Figure 1. Confirmed two-site cyber-range topology and the boundary between documented infrastructure and the proposed research overlay. Nominal provider rates are not measured throughput.

Table 1. Confirmed components and proposed research-overlay boundary.

Layer or site	Confirmed from supplied description	Not supplied or only proposed
Site A network	Keenetic Titan; 3 × 1 Gbit/s and 2 × 100 Mbit/s nominal WAN; 48 APs; CloudKey Gen2	Firmware; switches; VLAN/ACL/IP plan; uplinks for 36 APs; authorized export/API
Site A power	Three EcoFlow stations	Models; capacity; connected load; safety and switching procedure
Site B network	Keenetic Viva; 2 × 1 Gbit/s nominal WAN; 6 APs at 100 Mbit/s; CloudKey Gen1	Firmware; switches; VLAN/ACL/IP plan; authorized export/API
Site B compute	25 Kali Linux workstations	CPU/RAM/storage; installation rights; workload limits
Inter-site path	Protected VPN between sites	Protocol; cipher profile; MTU; routing; measured stable throughput
Research overlay	Reference code and synthetic artifact package	Deployed collectors; production keys; field twin service; distributed ledger; metaverse client

3.3. Assurance objectives and threat model

The protected assets are the admissible telemetry stream, current twin state, detector decision, explanation attached to that decision, and evidence required for reconstruction. The modeled adversary may modify a payload after signing, reuse an accepted identifier, replay an earlier signed event, alter the source sequence, break the previous-hash relationship, or submit an event under an invalid or unknown signature. The model also includes non-adversarial changes that may resemble attacks at the feature level: WAN failover, VPN degradation, a Wi-Fi authentication surge, and a backup-power transition. Scenario naming may be mapped to MITRE ATT&CK terminology where appropriate, but a knowledge-base mapping is not itself evidence that a defensive control is effective [23].

The integrity gateway assumes that legitimate sensor private keys remain secret and that the verifier has an authentic mapping from pseudonymous sensor identity to public key. Key theft is not solved by signature verification. A fully compromised authorized sensor may sign false but internally consistent values. Such semantic deception requires cross-source consistency, operational rules, anomaly detection, revocation, and investigation. Denial of service against ingestion, malicious time sources, kernel compromise, side-channel attacks, and physical manipulation of equipment were not exercised by the completed benchmark.

The reference code performs no scanning, exploitation, credential attack, or network reconfiguration. It accepts synthetic or exported telemetry and produces state and evidence records. The future field protocol excludes public targets, production identities, rogue access points, uncontrolled denial of service, and automatic remediation. The artifact is an observation and assurance component, not an autonomous offensive tool. **Table 2** maps each modeled disturbance to its implemented control and residual boundary.

Table 2. Threats, enforced controls, and residual risks.

Threat or disturbance	Implemented control or observation	Residual boundary
Modified signed payload	Canonical payload hash and Ed25519 verification	Compromised legitimate key can sign false values
Replayed event	Accepted-ID set, strictly increasing source sequence, previous-event hash	Durable state and key revocation require deployment engineering
Broken event order or chain	Sequence and previous-hash validation	Clock-source attacks are outside completed tests
Invalid/unknown identity	Registered public-key verification	Enrollment and rotation are not a production PKI
Operational anomaly	Static rules plus robust multivariate score	Field calibration and concept drift not established
Evidence modification	Previous-block hash, recomputed block hash, 3-of-4 signatures	Single-process implementation is not distributed consensus
WAN/VPN/Wi-Fi/power change	Planned scenario ground truth and topology-aware state	Physical behavior not yet measured

3.4. MetaTwin-Guard architecture

The artifact implements six logical stages. A software sensor first creates a canonical event containing source identity, source-local sequence, timestamps, topology node, feature payload, payload hash, previous-event hash, and signature. The integrity gateway then validates the schema, identifier, payload hash, Ed25519 signature, sequence monotonicity, and source-chain continuity. Static rules and the robust detector evaluate operational features. A hybrid decision combines their outputs with integrity status. Only an admitted event can update the twin-state store. Finally, a decision record is appended to a hash-chained evidence structure. **Figure 2** depicts this trust pipeline and its evidence path.

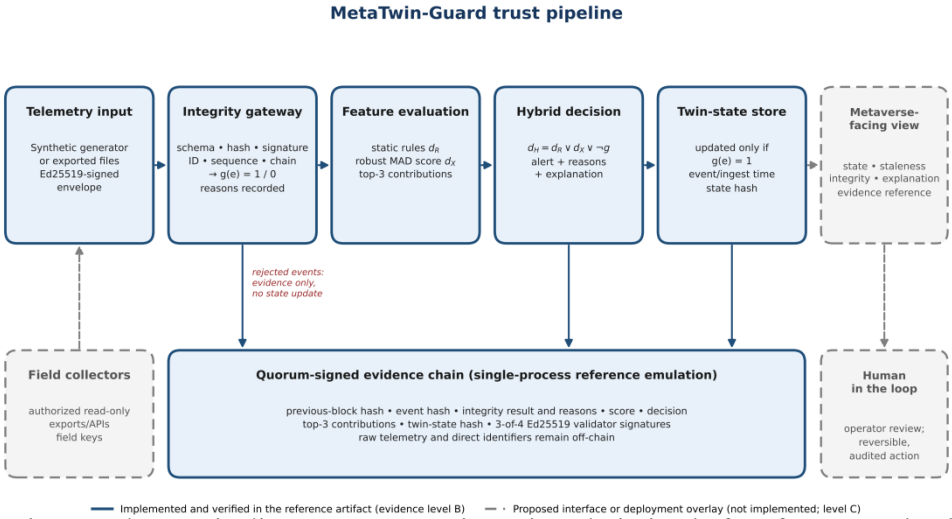


Figure 2. MetaTwin-Guard trust pipeline. Events pass integrity admission before feature evaluation and twin-state commitment; rejected events generate an evidence record but do not update the twin state. Solid blue outlines = components implemented and verified in the reference artifact (evidence level B); dashed grey outlines = proposed field collectors, metaverse-facing view, and human-in-the-loop action, which are not implemented (evidence level C). The evidence chain is a single-process emulation, not a distributed blockchain.

The current artifact operates at state and assurance layers. It does not claim to implement an immersive client, photorealistic model, industrial process simulator, or complete production digital twin. A future two-dimensional or three-dimensional interface may consume the verified state, staleness, integrity result, explanation, and evidence reference. This backend-first boundary is intentional: rendering a topology does not verify the state being rendered.

The signed event envelope includes event_id, scenario_id, phase, site_id, sensor_id, source_seq, event_time_utc, ingest_time_utc, topology_node, schema_version, a feature map, payload_hash, previous_hash, and signature. The synthetic generator uses deterministic private keys to make its output reproducible. Those keys are test fixtures and must never be reused in a field deployment.

3.5. Integrity admission and twin-state invariant

For incoming event e_i, the gateway validates required fields and types; recalculates the canonical payload hash; verifies the Ed25519 signature with the registered source key; confirms that the identifier has not previously been accepted; confirms that the source sequence is strictly greater than the last accepted sequence; and confirms that the supplied previous hash matches the last accepted event from that source.

Let C_{schema}, C_{hash}, $C_{\text{signature}}$, C_{id}, C_{sequence}, and C_{chain} be Boolean check results. Admission is

$$g(e_i) = C_{schema} \wedge C_{hash} \wedge C_{signature} \wedge C_{id} \wedge C_{sequence} \wedge C_{chain} \quad (1)$$

If any check fails, the event is rejected and its reasons are recorded. It is excluded from the state transition. For twin state T_i and deterministic update function U,

$$T_i = \begin{cases} U(T_{i-1}, e_i), & g(e_i) = 1 \\ T_{i-1}, & \text{otherwise} \end{cases} \quad (2)$$

This invariant is central to replay resistance. A replay may have plausible network values, but it cannot silently replace or refresh current state when it violates identifier, sequence, or source-chain constraints. Rejected events are still committed to the evidence chain so that attempted manipulation remains visible.

The invariant also determines behavior under delayed, missing, or interrupted telemetry. Absence of an admissible event never produces a state change: under

Equation (2) the last admitted state is held, and it is neither extrapolated nor reset to a nominal value. Holding state, however, creates the risk that an outdated state is read as current, so the twin must expose freshness explicitly. For topology node n at time t, staleness is defined as $\Delta_n(t) = t - t_{event,n}$, where $t_{event,n}$ is the event time of the last admitted update for that node. Two preregistered bounds classify the node as current ($\Delta_n \leq \tau_{fresh}$), stale ($\tau_{fresh} < \Delta_n \leq \tau_{lost}$), or unknown ($\Delta_n > \tau_{lost}$). A stale or unknown node must be rendered with its age and must not be shown with the visual encoding of a current healthy state. A delayed but valid event is admitted if its sequence number exceeds the last accepted one; it then updates the state with its own, older event time, so its staleness remains visible. An earlier event that arrives after a later one from the same source is rejected by the sequence check and recorded as evidence; legitimate out-of-order delivery therefore requires a bounded reordering buffer at the collector, which belongs to the field deployment rather than to the reference artifact. The reference artifact retains the event and ingest times of every admitted event, from which staleness can be computed; the freshness classification and its bounds are specified for the field protocol and the interface contract and were not exercised by the synthetic benchmark.

3.6. Robust explainable detector

For each feature j, the baseline median m_j and median absolute deviation MAD_j are computed from the baseline-phase events [24]. The implementation applies the normal-consistency factor 1.4826 and a uniform lower bound of 0.25 that prevents division by zero or unstable sensitivity when a baseline feature is nearly constant across heterogeneous units:

$$s_j = max\left(1.4826MAD_j, 0.25\right) \quad (3)$$

For new observation x, the absolute standardized deviation is clipped at 20:

$$z_j = min\left(20, \frac{|x_j - m_j|}{s_j}\right) \quad (4)$$

The multivariate anomaly score is the root mean square across p features:

$$A\left(x\right) = \sqrt{\frac{1}{p}\sum_{j=1}^{p} z_j^2} \quad (5)$$

The observation is flagged when $A(x) \geq 3.20$. Because the dispersion floor 0.25 is applied uniformly across heterogeneous units, a single binary transition ($|x_j - m_j| = 1$) yields $z_j = 4.0$, producing an isolated multivariate score of $A(x) = \sqrt{(16/7)} \approx 1.51$. Consequently, an isolated binary state change cannot breach the 3.20 threshold on its own (requiring at least five simultaneous deviations of $z = 4.0$ across the $p = 7$ features), which structurally accounts for the low robust-score response to power transitions. The fixed threshold and uniform floor represent an uncalibrated reference artifact configuration; prospective field protocols must evaluate per-feature scale normalization or dedicated rule fusion rather than post hoc parameter tuning.

The two constants were fixed before the benchmark as reference configuration values and were not tuned on the evaluation events. Their joint effect can be stated exactly. Because the score averages over $p = 7$ features, the threshold of 3.20 corresponds to a root-mean-square deviation of 3.20 robust units: a single continuous feature triggers an alert alone only if its standardized deviation reaches $3.20 \cdot \sqrt{7} \approx 8.47$, whereas two equal deviations each need about 5.99 and three about 4.89. The floor of 0.25 affects only features whose scaled baseline dispersion $1.4826 \cdot \mathrm{MAD}_j$ is below 0.25, that is, in the evaluated vector, chiefly the binary VPN and power indicators and any near-constant counter; continuous features such as RTT, jitter, and throughput remain standardized by their own dispersion. Because a uniform floor assigns the same weight to a unit change in features with different units, its consequences for binary indicators are analyzed in Section 4.2. That analysis shows that the floor, rather than the threshold, determines whether an isolated binary transition can be detected; feature-specific floors or explicit rule fusion are the

corresponding alternatives, and either must be fixed before confirmatory field runs.

The explanation is generated directly from the deviations used by the score. For nonzero total deviation, feature contribution c_j is

$$c_j = \frac{z_j}{\sum_{k=1}^{p} z_k} \quad (6)$$

The artifact reports the three largest contributions. This explanation is locally faithful to the implemented score: increasing a feature's robust deviation increases its contribution. It is not a causal explanation. A high RTT contribution does not prove that a provider caused an event, and multiple high contributions may be consequences of one latent fault.

The evaluated feature vector contains RTT in milliseconds, jitter in milliseconds, packet loss percentage, throughput in Mbit/s, authentication-failure count per window, binary VPN status, and binary power status. The field protocol treats unavailable modalities as missing rather than converting them to zero. Feature definitions, sampling windows, and measurement tools must be frozen before confirmatory collection.

3.7. Static rules and hybrid decision

The static comparator encodes seven transparent operational conditions: RTT > 180 ms; jitter > 50 ms; loss $> 5\%$; throughput < 8 Mbit/s; authentication failures > 18; VPN status < 0.5; or power status < 0.5. The Boolean rule detector is positive when any condition is true. These values are artifact configuration constants, not vendor limits or validated field thresholds. They were fixed before the benchmark as illustrative values for the synthetic generator's operating envelope and are not intended as generally applicable operational limits. In the field protocol, each rule threshold is to be re-derived from the pilot baseline of the corresponding approved path and frozen before confirmatory runs.

Let d_{R} be the static-rule output, d_{X} be the robust-score output, and g be the integrity-admission result. The hybrid decision is

$$d_H(e_i, x_i) = d_R(x_i) \vee d_X(x_i) \vee \neg g(e_i) \quad (7)$$

The OR construction ensures that an integrity rejection becomes an assurance alert even when feature values look normal. It may also increase false positives, so its F1 score is not guaranteed to exceed both components in every scenario. A future calibrated fusion model would be a separate method and must not replace the preregistered reference configuration without disclosure.

3.8. Quorum-signed evidence chain

QuorumEvidenceLedger is a single-process emulation of a permissioned, quorum-signed evidence structure. Every block contains its index, previous-block hash, event identifier, event-payload hash, integrity result and reasons, anomaly score, hybrid decision, top feature contributions, and current twin-state hash. Four deterministic Ed25519 validator identities sign each block hash; verification requires at least three valid signatures.

For block B_k, the block hash is computed from the canonical block body, including the previous-block hash. Verification recomputes the body hash, checks the chain relationship, and verifies the available registered signatures. The structure provides tamper evidence and a testable evidence format inside the reference process. It does not execute a peer-to-peer protocol, leader election, persistence replication, Byzantine agreement, or network consensus. The measured append time therefore reflects local hashing, serialization, and signing, not blockchain confirmation.

The design stores hashes and minimal decision metadata rather than raw telemetry on the evidence chain. This reduces disclosure and deletion risks. A production implementation would require independent keys, protected storage, key rotation and revocation, persistence, recovery, distributed-failure tests, and an explicit governance model. Work combining digital twins, smart contracts, and

explainable detection illustrates the integration opportunity, but production trust cannot be inferred from architectural composition alone [22].

The security properties of the quorum rule depend on assumptions that the reference emulation does not satisfy. The 3-of-4 rule tolerates compromise of up to two validator keys only if the keys are held by independent parties on separate hosts; an adversary controlling three keys can sign arbitrary blocks. In the present single-process emulation, compromise of the process exposes all four keys at once, so the quorum provides no protection against host compromise; its value in the artifact is to fix the evidence format and the verification rule. A production deployment would require validator keys generated and held on independent hosts, preferably in hardware-backed key stores; documented rotation and revocation; and recording of the validator set in each block so that historical blocks remain verifiable after rotation. For interrupted recording, a block should be acknowledged only after it has been fully signed and durably written. On restart, the chain is re-verified from the genesis block or from the last signed checkpoint, a partially written tail block is discarded, and any gap between the last committed block and the source sequence numbers is reported as an evidence gap rather than silently refilled. These recovery procedures, validator-key compromise, and process termination during append are neither implemented nor tested in the current artifact; they belong to the extended verification suite listed among the field-readiness gates in Section 4.4.

3.9. Deterministic synthetic verification

The benchmark used seed 20260823 and six scenarios: WAN failover, VPN degradation, Wi-Fi authentication surge, power transition, telemetry modification, and replay. Each scenario was repeated eight times. Each run contained 120 baseline events and 240 evaluation events, 40 of which were labeled injected anomalies and 200 of which were normal. The complete dry run therefore comprised 48 runs, 17,280 generated events, and 1,920 injected events.

The generator exercised different paths. WAN failover, VPN degradation, and Wi-Fi authentication surge primarily changed operational features. Power transition changed a low-dimensional binary state. Telemetry modification altered a signed event after its envelope had been created. Replay resubmitted an earlier event so that identity, sequence, or source-chain controls—not feature-space deviation—had to identify the problem. Labels originated from the deterministic generator; they were not inferred from physical logs. No records from external benchmarks such as CICIoT2023 were used [25]; consequently, these results verify the supplied implementation but do not establish cross-dataset generalization.

Three modes were evaluated from the same generated events: static rules, robust explainable score, and integrity-aware hybrid. Precision, recall, F1, false positives per 1,000 normal events, synchronization latency, evidence append time, and explanation top-three hit rate were recorded. Precision–recall measures are appropriate when positive events are less frequent than normal events, but no single scalar fully characterizes an imbalanced detector [26].

The artifact uses event-time and ingest-time fields to calculate a synthetic synchronization value. This is generator-defined time, not measured interface delivery or physical network latency. Both timestamps are assigned by the generator, so the reported synchronization value contains no software-processing delay of the artifact; processing cost appears only in the separately measured evidence-append time. Evidence append time was measured with the process monotonic clock. Eight deterministic repetitions assess reproducibility under the generator; they are not a population sample, and no inferential hypothesis test was applied. Because all repetitions derive from one declared master seed, they characterize run-to-run variation within one generator configuration rather than independent seeds. Additional master seeds, an executed ablation, and an executed sensitivity

analysis are required before the configuration is compared with alternatives; they are listed among the field-readiness gates in Section 4.4.

3.10. Prospective field experiment

The field phase may begin only after written authorization identifies responsible personnel, approved sites, address ranges, test services, dates, and stop authority. Experimental VLANs, SSIDs, identities, and targets must be separated from production. Egress from perturbation segments must be default-deny with an explicit allowlist. Configuration backups, an independent rollback path, test-only credentials, time synchronization, and protected field keys are mandatory. No scenario may rely on MetaTwin-Guard itself for safe termination. **Figure 3** presents the independently reset run as the field-experiment unit.

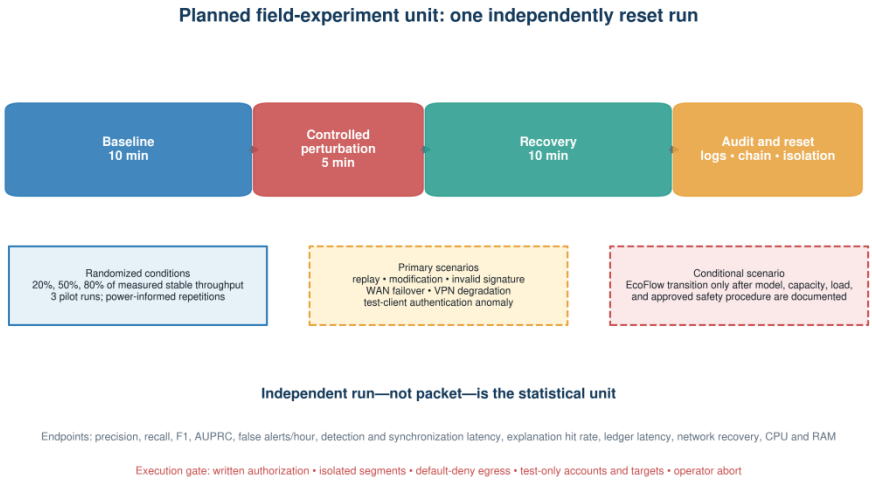


Figure 3. Planned field-experiment unit. One independently reset run—not an individual packet—is the statistical unit. Every run is gated by authorization, isolation, default-deny egress, test-only targets, and operator abort.

One planned run contains a pre-run health and isolation gate, 10 min of baseline observation, no more than 5 min of controlled perturbation, 10 min of recovery observation, and an audit/reset phase. Background load is assigned at 20% , 50% , or 80% of separately measured stable throughput for the approved path, not of nominal provider rates. Scenario and load order are randomized within day/site/path blocks. Matched normal-operation controls are interleaved. Three pilot runs are used only to verify safety, timing, completeness, and variance assumptions; confirmatory repetitions are determined by an a priori power analysis using run-level variance and a declared minimum relevant effect. **Table 3** enumerates the planned scenarios, endpoints, and execution gates.

Table 3. Planned field scenarios and execution gates.

ID	Planned condition	Primary endpoints	Execution boundary
S0	Matched normal-operation control	False alerts; synchronization; completeness; resource use	Isolated benign test traffic only
S1	Replay previously accepted signed test event	Rejection recall; quarantine latency; state invariance	Telemetry interface only; no production payload
S2	Modify test event after signing	Modification rejection; reason correctness; state invariance	Authoritative logs remain unchanged
S3	Invalid signature or unregistered test key	Rejection recall; false rejection; reason correctness	Pre-created test keys only
S4	Administratively controlled WAN failover	Detection; session interruption; failover and recovery time	No production/emergency traffic; independent rollback

Continued table:

ID	Planned condition	Primary endpoints	Execution boundary
S5	Controlled VPN delay/jitter/loss/rate limit	RTT; jitter; loss; throughput; detection and recovery	Approved test path only; no cryptographic downgrade
S6	Bounded test-client authentication anomaly	Precision; recall; F1; AUPRC; explanation hit rate	Test-only SSID/service and accounts
S7	Conditional EcoFlow transition	Availability; continuity; recovery	Execute only after model/load/electrical procedure approval

S1–S6 are the primary scenario set. S7 remains conditional and exploratory until equipment models, loads, protective limits, responsible personnel, and an electrical-safety procedure are documented. Any unexpected egress, production impact, loss of rollback access, device alarm, exceeded safety limit, uncontrolled propagation, credential exposure, or operator concern triggers an immediate stop. Invalid and safety-stopped runs remain in the accountability table and are never silently deleted or replaced.

Primary outcomes include precision, recall, F1, area under the precision–recall curve where a continuous score is available, false alerts per hour, integrity-rejection recall, false-rejection rate, detection latency, synchronization latency, explanation top-three hit rate, network recovery, evidence verification, CPU utilization, memory, and storage growth. Training, calibration, validation, and test partitions contain complete runs or days; events from a run are not split randomly across partitions.

For run *r*, planned detection latency is

$$L_{det,r} = t_{firstcorrectalert,r} - t_{injectionstart,r} \quad (8)$$

Runs with no alert inside the observation window are treated as right-censored, not assigned an artificial maximum. Synchronization latency for accepted event *i* is

$$L_{sync,i} = t_{twincommit,i} - t_{event,i} \quad (9)$$

This endpoint is valid only when clock offset remains within the preregistered tolerance. Confirmatory mode comparisons are paired within run. Confidence intervals resample runs as the highest-level cluster; an event-only bootstrap is not presented as if events were independent replicates. False-alert counts use exposure time, and right-censored latency is analyzed with appropriate survival or restricted-mean methods. Holm adjustment is planned within each confirmatory endpoint family.

The field protocol uses automated test clients and does not evaluate student or operator performance. If later work collects human decisions, usability, workload, identifiable activity, audio, video, or biometrics, collection requires a separate competent ethics determination and consent procedure where applicable. Public artifacts must exclude credentials, private keys, operational IP addressing, sensitive topology, personal data, and exploit-enabling configuration. The full prospective protocol and blank data schemas are supplied as supplementary information (**Supplementary Files S1 and S2**).

4. Results and Discussion

4.1. Automated artifact verification

The source code, configuration, unit tests, and benchmark outputs of the reference artifact are provided in **Supplementary File S3** (MetaTwin-Guard v1.0.1 [27]); how this release relates to the verification run reported below is stated in the Data and Code Availability statement. The reference package was evaluated under CPython 3.11.8 on an x86_64 workstation (Intel Core i7-8665U CPU @ 1.90 GHz, 16 GB RAM) running Ubuntu 22.04 LTS (Linux kernel 5.15). Cryptographic primitives were provided by the standard hashlib module (SHA-256) and the cryptography package (version 42.0.5) for Ed25519 signatures (RFC 8032).

Automated verification was invoked via python3 -m unittest discover tests. Five automated unit tests passed, covering acceptance of a clean signed stream and state update, replay rejection without a twin-state change, post-signature telemetry-tamper rejection, detection of tampering with a stored evidence block, and production of benchmark outputs with a verifiable ledger. Python bytecode compilation (python3 -m compileall .) also completed without error. **Table 4** records the tested invariants and outcomes.

Table 4. Automated verification of the reference artifact.

Test	Expected invariant	Result
Clean baseline stream	Valid events accepted; twin state updated	Pass
Replay event	Rejected; accepted state unchanged; evidence retained	Pass
Modified signed telemetry	Signature/hash inconsistency rejected	Pass
Evidence-block tampering	Verification reports the altered chain/block	Pass
Benchmark generation	CSV/metadata produced; evidence chain verifies	Pass

All benchmark runs reported a valid evidence chain. This result shows that the implemented verification function accepted the artifact-generated blocks and detected the tested block mutation. It does not establish persistence durability, validator independence, distributed fault tolerance, or field key security.

The five tests verify the core admission, state, and evidence invariants only and are not sufficient to establish broader system reliability. They do not exercise component failure, concurrent ingestion from several sources, crash recovery, interrupted evidence append, or security edge cases such as key revocation, clock manipulation, malformed canonical encodings, and identifier collisions across sources. These cases define an extended verification suite that must pass before any field deployment; it is listed as a separate field-readiness gate in Section 4.4 and is not represented as completed.

4.2. Synthetic detector behavior

Table 5 reports scenario-level mean performance and sample standard deviation over eight deterministic runs. Static rules had higher mean F1 than the robust detector for WAN failover, VPN degradation, power transition, and telemetry modification. The robust detector was marginally higher for Wi-Fi authentication surge. Neither feature-only mode detected replay, whereas the integrity-aware hybrid achieved mean F1 0.879. The differences follow from which signal each scenario perturbs: multi-feature degradations that cross explicit thresholds favor static rules, an isolated binary transition is diluted in the averaged robust score (Section 3.6), and replay leaves the feature vector unchanged.

Table 5. Deterministic synthetic dry-run performance (mean ± sample SD; eight runs per scenario). Evidence level B: synthetic verification, not field data.

Scenario	Mode	Precision	Recall	F1	False positives / 1,000 normal events
WAN failover	Rules	0.877 ± 0.029	0.953 ± 0.047	**0.913 ± 0.026**	26.875 ± 7.530
WAN failover	Robust score	0.787 ± 0.048	1.000 ± 0.000	0.880 ± 0.030	55.000 ± 15.584
WAN failover	Hybrid	0.787 ± 0.048	1.000 ± 0.000	0.880 ± 0.030	55.000 ± 15.584
VPN degradation	Rules	0.908 ± 0.039	0.916 ± 0.050	**0.911 ± 0.033**	18.750 ± 8.763
VPN degradation	Robust score	0.783 ± 0.050	1.000 ± 0.000	0.878 ± 0.031	56.250 ± 15.980
VPN degradation	Hybrid	0.783 ± 0.050	1.000 ± 0.000	0.878 ± 0.031	56.250 ± 15.980

Table 5. (Continued).

Scenario	Mode	Precision	Recall	F1	False positives / 1,000 normal events
Wi-Fi authentication surge	Rules	0.909 ± 0.036	0.738 ± 0.052	0.813 ± 0.037	15.000 ± 6.547
Wi-Fi authentication surge	Robust score	0.791 ± 0.041	0.847 ± 0.087	**0.815 ± 0.036**	45.625 ± 13.999
Wi-Fi authentication surge	Hybrid	0.791 ± 0.041	0.847 ± 0.087	**0.815 ± 0.036**	45.625 ± 13.999
Power transition	Rules	0.922 ± 0.052	1.000 ± 0.000	**0.959 ± 0.028**	17.500 ± 12.247
Power transition	Robust score	0.163 ± 0.106	0.047 ± 0.028	0.071 ± 0.042	48.750 ± 19.039
Power transition	Hybrid	0.808 ± 0.059	1.000 ± 0.000	0.893 ± 0.037	48.750 ± 19.039
Telemetry modification	Rules	0.950 ± 0.025	1.000 ± 0.000	**0.974 ± 0.013**	10.625 ± 5.630
Telemetry modification	Robust score	0.872 ± 0.048	1.000 ± 0.000	0.931 ± 0.028	30.000 ± 12.817
Telemetry modification	Hybrid	0.872 ± 0.048	1.000 ± 0.000	0.931 ± 0.028	30.000 ± 12.817
Replay	Rules	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	19.375 ± 9.797
Replay	Robust score	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	55.625 ± 19.899
Replay	Hybrid	0.787 ± 0.061	1.000 ± 0.000	**0.879 ± 0.038**	55.625 ± 19.899

Note: bold values mark the highest mean F1 in each scenario; tied values are both marked.

Figure 4 visualizes the scenario-level F1 comparison and its run-to-run variability without changing the synthetic-evidence boundary.

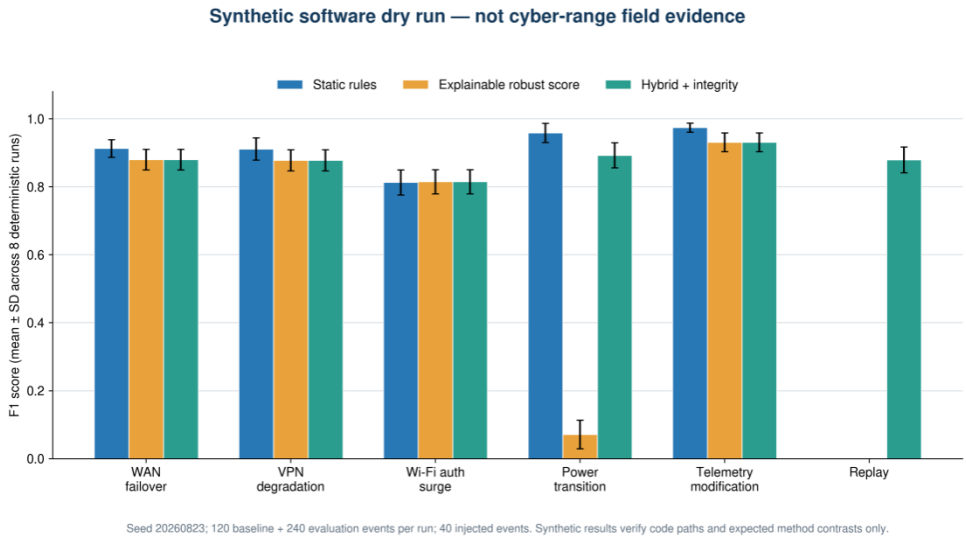


Figure 4. Mean F1 ± sample standard deviation across eight deterministic synthetic runs per scenario. These results verify code paths and expected method contrasts; they are not cyber-range field evidence.

The largest component-level difference occurred for power transition. Static rules obtained mean F1 0.959, while the robust detector obtained 0.071. The hybrid recovered to 0.893 because the explicit binary power rule supplemented the root-mean-square score. The result is consistent with score dilution: one binary feature may deviate strongly while the remaining features stay near baseline, reducing an average over p dimensions.

Replay demonstrates the distinct role of integrity admission. A replayed event may contain ordinary network values, so both feature-only methods returned zero F1. The hybrid detected replay through the integrity-rejection term. Telemetry modification produced high F1 in all modes, but the hybrid did not exceed static rules; its OR fusion admitted additional positives and therefore did not guarantee higher precision.

To separate the contribution of each layer, **Table 6** reports a derived ablation with two additional configurations: integrity-only ($d = \neg g$) and rules plus integrity ($d = d_R \vee \neg g$). These values were not obtained by re-executing the benchmark.

They follow from Equation (7) and the mean counts underlying **Table 5** (40 injected and 200 normal evaluation events per run), under two conditions confirmed by the automated tests: validly signed, fresh events are admitted, and replayed or post-signature-modified events are rejected. In WAN failover, VPN degradation, Wi-Fi authentication surge, and power transition, no event violates integrity, so integrity-only produces no alerts and rules plus integrity reduces exactly to static rules. For replay, a hybrid recall of 1.000 combined with zero recall of both feature-based modes implies that every replayed event was detected solely through the integrity term. Precision of the derived configurations is computed from mean counts; applied to the hybrid, this procedure gives a replay precision of 0.782 and F1 of 0.878, against reported per-run means of 0.787 and 0.879, which indicates the size of the approximation.

Table 6. Derived layer ablation (mean F1). Hybrid and component values are taken from Table 5; derived values follow from Equation (7) and are not re-executed results.

Scenario	Rules	Robust score	Hybrid (reported)	Integrity-only (derived)	Rules + integrity (derived)
WAN failover	0.913	0.880	0.880	0.000	0.913
VPN degradation	0.911	0.878	0.878	0.000	0.911
Wi-Fi authentication surge	0.813	0.815	0.815	0.000	0.813
Power transition	0.959	0.071	0.893	0.000	0.959
Telemetry modification	0.974	0.931	0.931	1.000	0.974
Replay	0.000	0.000	0.879	1.000	≈0.954

Three conclusions follow. First, the replay advantage of the hybrid is attributable entirely to the integrity layer: in that scenario the robust-score term contributes only false positives, and an integrity-only detector would reach F1 = 1.000. The hybrid result for replay must therefore not be read as evidence that statistical detection helps against replay. Second, in five of six scenarios the rules-plus-integrity configuration matches or exceeds the full hybrid, and the robust score improves F1 only for Wi-Fi authentication surge (0.815 versus 0.813), a difference far smaller than one run-level standard deviation. Third, the robust score remains a transparent indicator for multivariate patterns that no static rule encodes; in the present generator such patterns are rare, which is a property of the synthetic scenarios rather than evidence about field behavior. An executed ablation over additional master seeds is required to confirm these derived values.

Table 7 reports an analytical sensitivity analysis of the robust detector for binary transitions. For a binary feature with a constant baseline, the standardized deviation after a unit change is $z = \min(20, 1/s_{\min})$, and the minimum number k of simultaneous equal deviations that reaches threshold τ is $\lceil p\tau^2/z^2 \rceil$ with $p = 7$. These quantities follow directly from Equations (3)–(5) and require no re-execution.

Table 7. Analytical sensitivity of the robust score to the dispersion floor and threshold for an isolated binary transition (p = 7; "not reachable" means that more than seven features would be required).

Floor s_min	z after unit binary change	Isolated score A	Minimum k at τ = 2.5	Minimum k at τ = 3.2	Minimum k at τ = 4.0
0.10	10.0	3.78	1	1	2
0.25 (reference)	4.0	1.51	3	5	7
0.50	2.0	0.76	not reachable	not reachable	not reachable

With the reference floor, an isolated power or VPN transition cannot trigger the robust score at any of the examined thresholds, which explains the robust-score

F1 of 0.071 for power transition: its few detections (recall 0.047) can arise only when other features deviate concurrently. Lowering the floor for binary indicators to 0.10 would make an isolated transition detectable at $\tau = 3.20$, but the score would then act as a rule on that indicator and would become sensitive to any spurious toggling. Lowering the threshold to 2.5 does not resolve the problem with the reference floor and would increase false positives for continuous features. The field protocol therefore treats binary state indicators through explicit rules and evaluates feature-specific floors as a preregistered alternative rather than a post hoc adjustment. False-positive rates under alternative settings depend on the empirical baseline distribution and cannot be derived analytically; they must be reported from executed runs.

4.3. Synthetic timing and explanation outputs

Mean synthetic synchronization latency—computed from generator event and ingest timestamps—was 156.3 ms for WAN failover, 160.0 ms for VPN degradation, 45.8 ms for Wi-Fi authentication surge, 45.8 ms for power transition, 45.9 ms for telemetry modification, and 46.0 ms for replay. This separation reflects the generator's scenario-specific transport delay model: WAN failover and VPN degradation scenarios programmatically inject elevated network propagation and queueing offsets (nominal ~150–160 ms), whereas the remaining four scenarios reflect the baseline ingestion offset (~45–46 ms). These values quantify synthetic modeling assumptions rather than measured physical provider switching, VPN recovery, telemetry transport, or metaverse interface delivery.

In the delivered verification run, the scenario-level means of per-run median local append time rounded to 0.273–0.275 ms; the unrounded, environment-dependent values are retained in the supplementary CSV (**Supplementary File S3**). These sub-millisecond observations quantify local code execution only. They are not distributed quorum or blockchain confirmation times and may change when the benchmark is repeated on another host or at another time.

For scenarios evaluated by the robust detector, the mean top-three explanation hit rate was 1.000 for WAN failover, VPN degradation, and telemetry modification; 0.988 for Wi-Fi authentication surge; and 0.975 for power transition. The generator defines the changed feature set, so this metric shows alignment between the generator and the artifact's contribution calculation. It does not establish causal correctness or human usefulness in field conditions.

4.4. Field-readiness result

No physical field run had been executed at the time of writing. Consequently, the study reports no measured WAN switching time, VPN recovery time, wireless detection accuracy, EcoFlow autonomy, device resource use, human-response outcome, or real-network confidence interval. **Table 8** records the gating status rather than fabricated values.

Table 8. Field-readiness gates at manuscript preparation.

Gate	Status	Required resolution
Written range authorization and stop authority	Not supplied	Obtain signed scope, dates, operators, allowlists, and abort chain
Ethics/privacy determination	Not supplied	Institutional determination before any human/identifiable data collection
Network isolation and rollback evidence	Not supplied	Confirm experimental segments, default-deny egress, backup, and independent rollback
Complete device/software inventory	Partial	Record firmware, AP/switch models, VPN, hosts, and dependency versions
Authorized telemetry interfaces	Not confirmed	Use approved read-only exports/APIs or omit modality

Continued table:

Gate	Status	Required resolution
Production field keys and time synchronization	Not created/confirmed	Generate independent protected keys and document clock tolerance
Power-transition procedure	Conditional	Record EcoFlow model, capacity, load, electrical controls, and rollback
Pilot variance and confirmatory sample size	Not available	Run safety pilots, then perform a priori power analysis
Field observations and locked dataset	None	Execute protocol; audit and analyze by complete run
Extended reliability and security test suite	Not executed	Test component failure, concurrent ingestion, crash recovery, interrupted evidence append, and validator-key compromise
Multi-seed benchmark, executed ablation, and sensitivity runs	Not executed for the reported configuration	Repeat the benchmark with additional master seeds; execute integrity-only and rules-plus-integrity modes and alternative floors and thresholds

4.5. Implications for layered detection

The synthetic results support a layered design rather than exclusive reliance on a statistical score. Static rules performed strongly when a scenario directly crossed an explicit operational boundary. This was most apparent for power transition, where a binary power variable was sufficient for a deterministic rule but contributed only one dimension to the root-mean-square anomaly score. The explainable detector behaved consistently with its published equation; its low power-transition F1 identifies a design limitation rather than an unexplained implementation failure.

Wi-Fi authentication surge produced nearly equal F1 in the two feature-based methods, with the robust detector marginally higher. A robust baseline can express that an authentication count is unusual relative to local observations rather than only above a universal threshold. Field validation remains essential because schedules, client reconnects, controller maintenance, radio conditions, and software updates may create non-stationary patterns that are absent from the generator.

Replay shows why operational anomaly detection cannot replace cryptographic freshness and ordering. The feature vector need not change, so neither static thresholds nor deviation scoring is required to activate. The integrity gateway instead identifies the defect in event identity, sequence, or source-chain continuity. Authenticity, freshness, and anomaly are related but different properties. A trustworthy twin must represent all three rather than asking one model to infer them implicitly.

Telemetry modification illustrates the precision–recall trade-off of OR fusion. Static rules achieved the highest synthetic F1, while robust and hybrid modes were lower because their broader positive set increased false positives. Future field reporting should present precision, recall, AUPRC, false alerts per hour, run-level confusion counts, and operational consequences. F1 alone cannot show whether a mode fails through missed events or excessive alerts.

The practical role of the hybrid follows from these trade-offs. It is preferable where a missed manipulation costs more than additional review effort, for example, when the twin state feeds shared situational awareness or automated decision support and when replay, injection, or modification of telemetry is plausible. It is less suitable as a direct paging mechanism. In the synthetic runs, the hybrid inherited the robust-score false-positive rate of 30.0–56.3 per 1,000 normal events, compared with 10.6–26.9 for static rules. At an illustrative rate of one event per second per source, 55.6 false positives per 1,000 normal events would correspond to about 200 false alerts per source-hour, which would be operationally unacceptable without triage. Because the integrity result and the anomaly score are recorded separately from the hybrid decision, alerts can be tiered: integrity rejections as high-priority alerts that never update the state; rule violations as operational alerts; and detections by the robust score alone as advisory annotations that require corroboration from topology, a second source, or persistence over consecutive windows before escalation. This tiering is a recommendation for the field configuration and was not evaluated in the synthetic benchmark.

4.6. Meaning and limits of explanation

The top-three feature list is deliberately simple and fully determined by the robust deviations used in the score. It avoids claiming that a surrogate explains a separate opaque model. An operator can inspect whether a decision was associated primarily with RTT, jitter, loss, throughput, authentication failures, VPN status, or power status. This exact local consistency is useful for debugging and audit.

The explanation is not causal and may not be sufficient for action. A high loss contribution does not identify a failed provider, and a high authentication contribution does not distinguish malicious attempts from a controller or client fault. Several features can respond to one hidden cause, while a compromised legitimate

sensor can sign semantically false values. Explanations therefore need topology, integrity result, ground-truth timing in an experiment, and corroborating logs. The planned top-three hit metric assesses whether a manipulated feature appears among the reported contributors, not whether a human makes a better decision.

A representative decision trace illustrates how these elements combine; the values were computed with Equations (3)–(6) for illustration and are not benchmark records. Suppose an event from the inter-site VPN path node is admitted ($g = 1$) with standardized deviations of 6.0 for RTT, 5.2 for jitter, 4.4 for loss, 2.0 for throughput, 0.3 for authentication failures, and 0 for the VPN and power indicators. Its score is $A = 3.52$, above the threshold of 3.20, and the top-three contributions are RTT (0.34), jitter (0.29), and loss (0.25). If the raw RTT also exceeds the 180 ms rule, the alert combines an authenticated source, mutually consistent path features, and a single topology element, which together indicate path degradation rather than manipulation; the appropriate response is operational diagnosis of the VPN path. In contrast, a replayed event carrying ordinary values (all deviations ≤ 0.6, $A = 0.36$) produces no feature evidence, but the gateway rejects it with an identifier or sequence reason; the state is unchanged, and the alert points to a telemetry-integrity investigation rather than to the network. A third pattern, an admitted event with $z = 4.0$ on the power indicator and small deviations elsewhere ($A = 1.56$, power contribution 0.68), is detected by the power rule but not by the score; the explanation locates the change at the power node, and the low score shows that the event is isolated rather than a multi-symptom disturbance. Operators therefore distinguish disturbances from probable manipulation not from the explanation alone but from its combination with the integrity result, the topology node, and the consistency of features across neighboring nodes.

4.7. Evidence provenance and the blockchain boundary

The key state invariant is that a rejected event cannot update the twin. The evidence record connects the event hash, integrity reasons, decision, explanation, and state hash to a previous evidence block. Verification across all synthetic runs confirms consistency of this implemented format and the tested 3-of-4 signature rule.

The term "quorum-signed" must not be confused with "distributed." All four validators are deterministic keys instantiated in one process. The local append time mainly quantifies hashing, object construction, and signature operations. A networked permissioned ledger would add transport, persistence, peer validation, ordering, retries, and consensus. It could fail under partitions, correlated compromise, or loss of quorum. Claims of decentralized trust, Byzantine fault tolerance, or blockchain throughput require a separate distributed implementation and experiment.

The current structure nevertheless has practical value. It defines a minimal evidence object, verification rule, and connection between detection and state that a distributed implementation would need to preserve. It also avoids placing raw telemetry or personal identifiers in an append-only record. Whether a production system requires a distributed ledger at all should be decided against a threat model and governance needs; a protected centralized log with hardware-backed signing may be more appropriate in some deployments.

4.8. Validity threats and limitations

The principal limitation is the absence of field data. All F1, timing, and explanation values originate from a deterministic generator with one declared seed. Engineered anomalies are less diverse than real faults and attacks and may favor assumptions encoded by the feature transformations. Eight repetitions per scenario demonstrate repeatable generator variation but are not independent draws from an operational population. No inferential population claim is made. Public datasets such as CICIoT2023 [25] were not used because they lack signed envelopes, source

sequences, topology nodes, and power-state fields; replaying such traces through a signing collector would test the detectors but not the integrity path. All conclusions are therefore restricted to software-level verification of the reference artifact.

The robust detector models each feature through a baseline median and MAD and aggregates deviations without learning temporal dependencies or covariance. Its fixed threshold and the static rules have not been calibrated on the physical range. Real wireless occupancy, scheduled maintenance, route asymmetry, controller upgrades, and time-varying load may shift distributions. The field protocol freezes an initial configuration, separates calibration from confirmatory runs, and reports any post hoc tuning as exploratory.

The evidence layer is a single-process reference implementation. Its validators do not fail independently, and the benchmark does not test durable storage, network partitions, recovery, key compromise, or distributed consensus. Local append timing cannot be cited as blockchain performance.

Infrastructure gaps also constrain reproducibility. Missing items include router and controller versions, access-point and switch models, VLAN/ACL/NAT topology, VPN details, EcoFlow models and loads, Kali workstation resources, authorized APIs or exports, and a confirmed Site A compute host. Until these are resolved, symmetric edge inference, two-site validator distribution, and path-capacity claims would be speculative.

Finally, the study assesses software decisions, not operator performance. No conclusion can be drawn about whether users understand explanations, respond faster, or make better decisions in a metaverse interface. Those questions require a separately designed human-participant study, competent ethics review, appropriate consent, and validated human-factors instruments.

5. Recommendations

Metaverse-oriented infrastructure operations require more than three-dimensional rendering. A user should know when a state was observed, which pseudonymous source supplied it, whether the event was admitted, which features made it unusual, how stale the rendered state is, and which evidence record supports it. MetaTwin-Guard addresses this backend assurance problem at the level of a verified software artifact. A future client can render state confidence, integrity status, explanation, and evidence reference without treating visual presence as proof.

The design also limits automation risk. The reference artifact generates decisions and evidence but does not isolate users, reroute production traffic, disable accounts, or control physical equipment. Human approval should remain mandatory until field measurement establishes acceptable false-alert behavior and failure modes. The same evidence can support operator review, reconstruction, and comparison between what infrastructure reported and what the interface displayed.

The documented range is a credible training and research environment because it combines two sites, multiple provider links, managed wireless infrastructure, a VPN, backup power, and test workstations. It is not evidence of an operational industrial or other regulated critical-infrastructure object. The contribution is therefore a cyber-range assurance backend and protocol that may later support emulation of critical-infrastructure scenarios. Generalization to energy, transport, manufacturing, or public-sector operational technology requires domain-specific sensors, hazards, controls, and validation.

The physical cyber range provides a relevant future environment through two sites, seven nominal provider connections, an inter-site VPN, 54 access points, backup-power nodes, and 25 Kali Linux workstations. The field experiment remains prospective. The next steps are to complete infrastructure inventory and isolation gates, obtain institutional authorization, execute pilot and preregistered confirmatory runs, and report physical measurements separately from synthetic

artifact verification.

6. Conclusion

This work defines an explainable and verifiable digital-twin assurance backend for a documented two-site cyber range. The artifact combines Ed25519 event authentication, schema and hash-chain admission controls, explicit operational rules, a transparent median/MAD robust score, integrity-aware hybrid decisions, protected state transitions, and a quorum-signed evidence chain.

Deterministic verification covered six scenarios, eight repetitions per scenario, and 17,280 generated events. Static rules were effective for explicitly thresholded operational changes, while the robust detector supplied transparent multivariate deviations. Replay exposed the limitation of feature-only detection: both operational modes yielded zero F1, whereas the integrity-aware hybrid detected the scenario. All synthetic evidence chains verified. The synchronization and append timings are software dry-run values only.

A derived layer ablation shows that the replay advantage of the hybrid is due entirely to the integrity layer, and an analytical sensitivity analysis shows that the uniform dispersion floor, not the threshold, prevents the robust score from detecting isolated binary transitions. The contribution of the work lies in the admission-before-state invariant and in evidence records that bind the integrity outcome, the explanation, and the twin-state hash, rather than in any individual mechanism. The study demonstrates software-level verification only; it does not show field performance, distributed-ledger behavior, or benefit to metaverse users, which require the prospective protocol and separate studies.

Author contributions: Conceptualization, O.K. and D.P.-T.; methodology, D.P.-T. and K.D.; software, D.P.-T. and A.H.; validation, M.A., Y.Z., and A.H.; formal analysis, O.K. and K.D.; investigation, D.P.-T. and M.A.; resources, O.K. and Y.Z.; data curation, A.H. and D.P.-T.; writing–original draft preparation, D.P.-T., K.D., and A.H.; writing–review and editing, O.K., M.A., and Y.Z.; visualization, D.P.-T. and A.H.; supervision, O.K.; project administration, D.P.-T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments and AI-use disclosure: Generative artificial intelligence tools were utilized strictly for illustrative figure rendering and language refinement (copy-editing) during the preparation of this manuscript. All conceptual formulations, software artifact engineering, methodological frameworks, experimental designs, and analytical evaluations represent the entirely original work of the authors, who maintain full responsibility for the contents and integrity of the final study.

Conflicts of interest: The authors declare no conflicts of interest.

Ethics statement: The completed deterministic software verification used synthetic events and involved no human participants. No ethics approval is claimed. Execution of the field protocol requires written cyber-range authorization and a competent institutional ethics/privacy determination before any human, operator-performance, or identifiable activity data are collected.

Data and code availability: The delivery package includes the executable reference code, unit tests, configuration, deterministic benchmark metadata, run-level CSV, summary CSV, and blank field-data schema. No field dataset exists. All of these, together with the figures, a Google Colab notebook with step-by-step demonstrations, and Supplementary Files S1 and S2, are archived on Zenodo as the versioned release MetaTwin-Guard v1.0.1 under the MIT license [27]: https://doi.org/10.5281/zenodo.23557608. The release documents the unit-test command (python3 -m unittest discover tests) and the benchmark command with seed 20260823, requires Python 3.10 or later and cryptography 42.0.5 or later, and

records the execution environment of every benchmark run; the results reported in this article were obtained under CPython 3.11.8 and cryptography 42.0.5. All values reported in this article (**Tables 4–7**, **Figure 4**, and the timing and explanation values in Section 4.3) come from the delivered verification run described in Sections 3.9 and 4.1, or are derived from it analytically; the run-level and summary CSV files of that run are part of the release. The synthetic generator in the release code is an independent implementation of the design specified in Section 3.9, so re-executing the benchmark with seed 20260823 does not reproduce that run exactly: the per-scenario mean F1 values differ from **Table 5** by at most 0.03, and all qualitative findings are reproduced. The executed layer ablation, sensitivity analysis, and additional-seed runs contained in the release were produced with this generator; they are provided for transparency, are not used for any value or conclusion in this article, and do not close the corresponding gate in **Table 8**, which refers to the configuration reported here. Operational addressing, credentials, keys, and sensitive topology are not included.

Supplementary materials: Supplementary File S1: prospective field experiment protocol. Supplementary File S2: blank field-data workbook and data dictionary. Supplementary File S3: MetaTwin-Guard v1.0.1 source code, configuration, tests, and deterministic synthetic benchmark outputs, archived at https://doi.org/10.5281/zenodo.23557608 [27]. Figures are supplied as 360 dpi PNG and editable SVG.

References

[1] International Organization for Standardization and International Electrotechnical Commission. ISO/IEC 30173:2023 Digital twin—Concepts and terminology. 2023. Available online: https://www.iso.org/standard/81442.html (accessed on 23 August 2026).

[2] International Organization for Standardization. ISO 23247-1:2021 Automation systems and integration—Digital twin framework for manufacturing—Part 1: Overview and general principles. 2021. Available online: https://www.iso.org/standard/75066.html (accessed on 23 August 2026).

[3] Somers RJ, Douthwaite JA, Wagg DJ, et al. Digital-twin-based testing for cyber–physical systems: A systematic literature review. Information and Software Technology. 2023;156:107145. https://doi.org/10.1016/j.infsof.2022.107145

[4] Wang Y, Su Z, Zhang N, et al. A Survey on Metaverse: Fundamentals, Security, and Privacy. IEEE Communications Surveys & Tutorials. 2023;25(1):319–352. https://doi.org/10.1109/COMST.2022.3202047

[5] National Institute of Standards and Technology. The NIST Cybersecurity Framework (CSF) 2.0. NIST CSWP 29; 2024. https://doi.org/10.6028/NIST.CSWP.29

[6] Ross R, Pillitteri V, Graubart R, et al. Developing Cyber-Resilient Systems: A Systems Security Engineering Approach. NIST SP 800-160 Vol. 2 Rev. 1; 2021. https://doi.org/10.6028/NIST.SP.800-160v2r1

[7] Nelson A, Rekhi S, Souppaya M, Scarfone K. Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile. NIST SP 800-61 Rev. 3; 2025. https://doi.org/10.6028/NIST.SP.800-61r3

[8] Tabassi E. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1; 2023. https://doi.org/10.6028/NIST.AI.100-1

[9] Sarhan M, Layeghy S, Portmann M. Evaluating Standard Feature Sets Towards Increased Generalisability and Explainability of ML-Based Network Intrusion Detection. Big Data Research. 2022;30:100359. https://doi.org/10.1016/j.bdr.2022.100359

[10] Barnard P, Marchetti N, DaSilva LA. Robust Network Intrusion Detection Through Explainable Artificial Intelligence (XAI). IEEE Networking Letters. 2022;4(3):167–171. https://doi.org/10.1109/LNET.2022.3186589

[11] Josefsson S, Liusvaara I. Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032; 2017. https://doi.org/10.17487/RFC8032

[12] National Institute of Standards and Technology. Secure Hash Standard (SHS). FIPS PUB 180-4; 2015. https://doi.org/10.6028/NIST.FIPS.180-4

[13] NICE Community Coordinating Council, Cybersecurity Skills Competitions Community of Interest. The Cyber Range: A Guide. National Institute of Standards and Technology; 2023. Available online: https://www.nist.gov/document/cyber-range (accessed on 23 August 2026).

[14] Chouliaras N, Kittes G, Kantzavelou I, et al. Cyber Ranges and TestBeds for Education, Training, and Research. Applied Sciences. 2021;11(4):1809. https://doi.org/10.3390/app11041809

[15] Vielberth M, Glas M, Dietz M, et al. A Digital Twin-Based Cyber Range for SOC Analysts. In: Data and Applications Security and Privacy XXXV. Springer; 2021. pp. 293–311. https://doi.org/10.1007/978-3-030-81242-3_17

[16] Rebecchi F, Pastor A, Mozo A, et al. A Digital Twin for the 5G Era: the SPIDER Cyber Range. In: 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM). IEEE; 2022. pp. 567–572. https://doi.org/10.1109/WoWMoM54355.2022.00088

[17] Zhou H, Li M, Sun Y, et al. Digital Twin-Based Cyber Range for Industrial Internet of Things. IEEE Consumer Electronics Magazine. 2023;12(6):66–77. https://doi.org/10.1109/MCE.2022.3203202

[18] Mathur AP. Reconfigurable Digital Twin to Support Research, Education, and Training in the Defense of Critical Infrastructure. IEEE Security & Privacy. 2023;21(4):51–60. https://doi.org/10.1109/MSEC.2023.3281272
[19] Park M, Lee H, Kim Y, et al. Design and Implementation of Multi-Cyber Range for Cyber Training and Testing. Applied Sciences. 2022;12(24):12546. https://doi.org/10.3390/app122412546
[20] Fu Y, Han W, Yuan D. Disentangled Orchestration on Cyber Ranges. IEEE Transactions on Dependable and Secure Computing. 2024;21(4):2344–2360. https://doi.org/10.1109/TDSC.2023.3303888
[21] Sanz-Rodrigo M, Rivera D, Moreno JI, et al. Digital Twins for 5G Networks: A Modeling and Deployment Methodology. IEEE Access. 2023;11:38112–38126. https://doi.org/10.1109/ACCESS.2023.3267548
[22] Kumar R, Aljuhani A, Javeed D, et al. Digital Twins-enabled Zero Touch Network: A smart contract and explainable AI integrated cybersecurity framework. Future Generation Computer Systems. 2024;156:191–205. https://doi.org/10.1016/j.future.2024.02.015
[23] MITRE. ATT&CK Enterprise Matrix. Available online: https://attack.mitre.org/matrices/enterprise/ (accessed on 23 August 2026).
[24] Rousseeuw PJ, Croux C. Alternatives to the Median Absolute Deviation. Journal of the American Statistical Association. 1993;88(424):1273–1283. https://doi.org/10.1080/01621459.1993.10476408
[25] Neto ECP, Dadkhah S, Ferreira R, et al. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. Sensors. 2023;23(13):5941. https://doi.org/10.3390/s23135941
[26] Saito T, Rehmsmeier M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLOS ONE. 2015;10(3):e0118432. https://doi.org/10.1371/journal.pone.0118432
[27] Korchenko O, Prokopovych-Tkachenko D, Davydenko K, et al. MetaTwin-Guard v1.0.1: Explainable and Verifiable Digital-Twin Assurance Backend for a Two-Site Cyber Range —Reference Software, Synthetic Benchmark, Colab Notebook and Field Protocol [Software]. Zenodo; 2026. https://doi.org/10.5281/zenodo.23557608