

**Dmytro Prokopovych-Tkachenko**

Candidate of Technical Sciences, Associate Professor;

Head of the Department of Cybersecurity and Information Technologies

University of Customs and Finance, Dnipro, Ukraine

ORCID: 0000-0002-6590-3898

*omega2417@umsf.edu.ua***CAUSAL-LAG NEURAL EVIDENCE FUSION FOR TEMPORAL TELEMETRY ATTACKS**

Abstract. Streaming telemetry can remain cryptographically valid while its temporal meaning is corrupted by clock drift, replay, selective delay, or low-and-slow cross-layer manipulation. This paper proposes Causal-Lag Neural Evidence Fusion (CL-NEF), a specialized detection method that treats event time and causal order as first-class security evidence. The method jointly processes source and receive timestamps, sequence and epoch identifiers, synchronization uncertainty, freshness, policy version, and context predicates from access, TCP/QUIC, SDN, and application layers. A robust affine clock estimator and online change-point engine produce uncertainty-bounded residuals; an admissible-lag graph then constrains a temporal graph-attention model with per-layer GRU-D encoders. An evidential output head separates gradual or stepwise clock drift, replay, delayed/reordered observations, slow cross-layer manipulation, and legitimate operating-mode changes while retaining an explicit epistemic-uncertainty signal. Evidence is accepted, marked suspect, quarantined, or rejected through a hysteretic state machine with time-to-live, bounded scope, and auditable rollback. The V2-V3 evaluation protocol uses a real messaging stack, SDN/NFV components, edge nodes, TCP/QUIC profiles, controlled synchronization error, observation loss, and gateway compromise. Resource-fair baselines comprise signature validation, a static freshness window, a univariate change detector, and a non-causal multivariate neural detector. F1, false-positive rate during legitimate mode changes, detection delay, calibration error, missing-observation robustness, and threshold transferability are reported on clearly labelled synthetic data produced by a seeded generator (20 seeds per cell) that encodes the author's design expectations; no field measurement is claimed, and a staged field trial with pre-registered go/no-go gates is proposed to replace every synthetic value. The design operationalizes a clear boundary between admissible telemetry evidence and quarantine without equating a valid signature with a temporally trustworthy state assertion.

Keywords: clock drift; replay attack; low-and-slow manipulation; streaming telemetry; causal-lag graph; graph attention; GRU-D; evidential deep learning; TCP; QUIC; SDN; edge computing; synthetic evaluation; field trial.

INTRODUCTION

Communication systems increasingly depend on streaming telemetry to reconstruct operational state across access networks, encrypted transports, programmable forwarding, message brokers, and application services. The security of this evidence is often reduced to integrity and authenticity: a record is considered trustworthy when its signature verifies, its source identity is known, and its timestamp falls inside a freshness window. That reasoning is insufficient when an attacker compromises a legitimate source, replays correctly signed records, introduces selective delay, or changes the source clock slowly enough to remain statistically plausible. The record can be authentic as a byte sequence while being false as a statement about the current system state.

Clock-related manipulation is particularly difficult in heterogeneous telemetry. Source timestamps are generated by devices with different oscillators, synchronization protocols, sleep states, and queueing paths. Receive timestamps include transport delay and collector scheduling. A static threshold cannot simultaneously tolerate benign synchronization error and expose gradual drift. The same ambiguity affects replay: a valid record may be duplicated within an apparently reasonable time interval, may cross an epoch boundary, or may be combined with fresh observations from other layers to create a coherent-looking but causally impossible narrative. Slow cross-layer manipulation further avoids point-anomaly detectors by changing several features below their individual alarm thresholds while progressively violating the lag structure that normally connects access, TCP/QUIC, SDN, and application events.



Prior work has established that clock skew can fingerprint remote devices [1,2], that synchronization requires explicit resilience assumptions [3], and that precision-time infrastructure is vulnerable to cyber manipulation [4-6]. Replay detection has also been formulated as a dynamic-delay estimation problem [7]. Separately, change-point analysis and deep multivariate anomaly detectors model temporal or inter-variable dependence [8-14], while causal discovery provides principled tests for lagged associations in nonlinear time series [15]. These lines do not yet provide a deployable mechanism that combines synchronization uncertainty, cross-layer causal admissibility, calibrated neural evidence, and a bounded quarantine decision for streaming communication telemetry.

Central research gap and research questions

The central gap is the absence of a detector that remains sensitive to temporally deceptive but cryptographically valid telemetry while remaining conservative during legitimate mode changes and uncertain synchronization. CL-NEF addresses this gap by constraining neural fusion with an explicit causal-lag graph and by converting epistemic uncertainty into an operational evidence state rather than forcing a binary anomaly decision.

RQ1: Can uncertainty-bounded temporal evidence distinguish gradual/step clock drift, replay, selective delay, and low-and-slow cross-layer manipulation when signatures remain valid?

RQ2: Does constraining neural attention by admissible causal-lag windows reduce false positives during legitimate route, transport, load, and policy changes?

RQ3: How transferable are learned scores and quarantine thresholds across wired TCP, Wi-Fi QUIC, congested edge, and partially observed profiles?

RQ4: At what synchronization error and observation-loss level does temporal evidence become too ambiguous to support acceptance and therefore require quarantine or abstention?

Contributions

1. A formal evidence-time representation that separates source time, receive time, clock correction, synchronization uncertainty, freshness, sequence/epoch continuity, and policy-version consistency.

2. A hybrid CL-NEF architecture combining robust drift estimation, online change-point evidence, per-layer missingness-aware recurrent encoders, admissible-lag graph construction, temporal graph attention, and an evidential uncertainty head.

3. A four-state accept-suspect-quarantine-reject policy with hysteresis, TTL, bounded scope, audit provenance, and controlled release or rollback.

4. A V2-V3 experimental protocol with resource-fair baselines, attack/benign scenario pairing, ablation, calibration, threshold-transfer testing, and explicit negative-result criteria.

RELATED WORK AND POSITIONING

Clock evidence and synchronization attacks

Remote clock skew has been used as a stable physical-device feature [1], and clock-skew analysis has supported detection of unauthorized wireless access points [2]. These studies demonstrate that timing residuals contain security information, but fingerprinting and device attribution are not equivalent to determining whether a current telemetry statement is temporally admissible. Secure synchronization research has formalized resilience against malicious participants [3], whereas later PTP-focused studies showed that practical time-distribution networks require attack-detection extensions and deployment-specific safeguards [4,5]. Multi-clock cooperation has recently been explored to improve time synchronization under attack [6]. CL-NEF treats these mechanisms as sources of uncertainty bounds and residual evidence rather than assuming a perfect global clock.

Replay, change points, and multivariate anomaly detection

Replay detection based on estimated dynamical delay can expose repetitions that preserve otherwise valid control data [7]. Relative density-ratio estimation provides a distribution-sensitive change-point mechanism without requiring explicit density estimation [8]. Neural multivariate detectors such as OmniAnomaly [9], MSCRED [10], MTAD-GAT [11], GDN [12], and TranAD [13] capture stochastic, convolutional, graph, or transformer structure. GRU-D explicitly represents informative missingness and irregular intervals [14]. These models are valuable baselines, but unconstrained correlation learning can accept a statistically familiar sequence that is impossible under known protocol and cross-layer lag semantics. Causal association analysis [15] motivates explicit lag-conditioned edges, yet operational telemetry security also needs uncertainty calibration and an enforcement state.

**Positioning of CL-NEF**

Table 1

Positioning against adjacent research lines.			
Research line	Primary evidence	Typical limitation for this problem	CL-NEF extension
Signature / MAC validation	Cryptographic authenticity	Cannot detect validly signed replay or compromised-source timing deception	Keeps authenticity as one predicate, not the final trust decision
Static freshness window	Absolute timestamp age	Conflates benign clock error with attacks; misses slow drift inside the window	Uses uncertainty-bounded evidence time and residual trends
Clock-skew / PTP attack detection [1-6]	Clock residual or synchronization message	Often single-layer and not linked to service-level causal order	Fuses clock evidence with TCP/QUIC, SDN, and application events
Replay delay estimation [7]	Estimated dynamical delay	Limited context for cross-layer partial replay and policy epochs	Adds sequence/epoch, policy version, graph lag, and calibrated abstention
Multivariate neural anomaly detection [9-14]	Learned statistical dependence	May learn spurious correlations and over-alarm on legitimate mode changes	Masks graph attention by admissible causal-lag and context predicates
Causal time-series analysis [15]	Lagged conditional association	Not directly an online evidence-handling policy	Maps causal inconsistency and uncertainty to bounded quarantine states

Source: compiled by the author on the basis of [1–15].

The proposed novelty is therefore not a claim that graph attention alone detects every temporal attack. It is the integration of explicit temporal admissibility, neural representation learning, uncertainty-aware evidence fusion, and an operational quarantine boundary that can be tested component by component.

SYSTEM SCOPE, TELEMETRY EVIDENCE, AND THREAT MODEL

The monitored communication system contains four evidence layers: access/edge, TCP or QUIC transport, SDN/forwarding, and application/messaging. An event record e_i includes a source identifier, layer, source timestamp, receive timestamp, sequence number, epoch, synchronization-uncertainty estimate, payload-independent features, policy version, and cryptographic-verification status. Records are processed online in overlapping windows; the method does not require payload decryption and can operate on protocol metadata, broker state, controller events, and signed service telemetry.

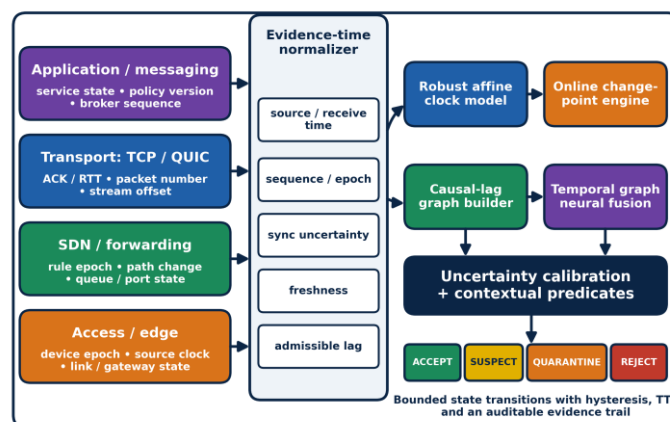
Cross-layer evidence architecture for clock-aware streaming telemetry analysis

Figure 1. Cross-layer evidence architecture for clock-aware streaming telemetry analysis. The diagram defines the proposed processing path and is not an empirical result.

Source: developed by the author using Python 3.12 / Matplotlib (architectural diagram of the proposed pipeline).



Table 2

Threat model, defender observations, and safety boundaries.

Element	Assumption / capability	Boundary of the claimed method
Attacker keys	May control a legitimate source or gateway and therefore emit correctly signed records	No claim that signature failure is required for detection
Timing actions	Gradual or step drift, replay, duplication, delay, reordering, and cross-epoch reuse	Clock correction is bounded; arbitrary global-time recovery is not assumed
Cross-layer actions	Coordinates sub-threshold changes across access, transport, SDN, and service telemetry	At least one uncompromised or independently observed evidence path is required
Network conditions	Congestion, route change, loss, retransmission, migration, and partition can be benign	The detector must model legitimate context before assigning attack evidence
Defender view	Receives timestamps, sequence/epoch, sync status, policy version, flow and service metadata	Payload content and perfect observability are not required
Safety action	Can quarantine evidence or a bounded source/flow scope	No autonomous destructive action; every state has TTL, audit trail, and release criteria

Source: developed by the author; assumptions generalize the attack capabilities reported in [3–7].

TEMPORAL AND CROSS-LAYER EVIDENCE FORMALIZATION

Evidence time under uncertain synchronization

For layer l , the receive clock is related to the source clock by a locally affine model over a bounded window. The parameters are not interpreted as permanent device fingerprints; they are online nuisance variables whose residuals may carry attack evidence. A robust estimator limits the leverage of delayed or replayed observations and produces an uncertainty interval rather than a single corrected timestamp.

$$t_{r,i} = a_l(t_{s,i}) + b_l(t_{s,i}) + \epsilon_{i,l} \quad (1)$$

$$r_{i,l} = t_{r,i} - (a_{\hat{l}} t_{s,i} + b_{\hat{l}}); \quad L_{i,l} = [t_{\hat{l}} - u_{i,l}, t_{\hat{l}} + u_{i,l}]. \quad (2)$$

Here $r_{i,l}$ is the clock residual and $L_{i,l}$ is the admissible evidence-time interval after correction. The uncertainty $u_{i,l}$ aggregates the reported synchronization error, estimator covariance, path-jitter allowance, and a bounded collector delay. Freshness is therefore evaluated against an interval: evidence is certainly fresh, ambiguous, or certainly stale. This avoids forcing a false anomaly when the corrected timestamp overlaps the permitted window only because uncertainty is large.

Sequence, epoch, policy, and causal-lag predicates

A record is not temporally admissible merely because its time interval is recent. Sequence continuity, epoch membership, and policy version define replay-sensitive predicates. Cross-layer causality is represented by a directed edge $j \rightarrow i$ only when the predecessor and successor types are compatible and their interval-valued lag intersects the configured admissible lag window.

$$A_{ji} = 1_{\text{type } 1_{\text{lag } 1_{\text{context}}}; (L_{i,l} - L_{j,l}) \text{ intersects } [L^{\wedge}_{-ji}, L^{\vee}_{+ji}]} \quad (3)$$

$$v_{i,l} = 1 - \max_j (A_{ji} q_{ji}), \quad q_{ji} \text{ in } [0, 1]. \quad (4)$$

The score $v_{i,l}$ summarizes causal-order violation after weighting each candidate predecessor by evidence quality q_{ji} . A missing predecessor does not automatically imply attack: the model distinguishes an observed contradiction from an unobserved edge. The missingness mask and elapsed time since the last observation are therefore supplied to the neural encoder.

Table 3

Core notation used by CL-NEF.

Symbol	Meaning	Operational source
$t_{s,i} / t_{r,i}$	Source and receive timestamps of event i	Endpoint, gateway, collector
a_l, b_l	Local clock-rate and offset parameters for layer/source l	Robust affine estimator



Symbol	Meaning	Operational source
u_i	Total synchronization and transport uncertainty bound	Sync daemon, estimator, jitter envelope
$seq_i / epoch_i$	Sequence and reset/restart domain	Protocol, broker, device session
pi_i	Policy/configuration version attached to the event	SDN controller, service orchestrator
$[L^-_{ji}, L^+_{ji}]$	Admissible lag from evidence type j to type i	Training data plus protocol constraints
m_i / δ_i	Observation mask and elapsed time since last value	Streaming window builder
v_i	Cross-layer causal-order violation score	Causal-lag graph
U_i	Epistemic uncertainty of the evidential head	Dirichlet evidence output
s_i	Evidence state: accept, suspect, quarantine, reject	Bounded decision state machine

Source: developed by the author (notation of the proposed method).

Detection target and non-goals

The target labels are benign mode change, clock drift, replay, delay/reordering, and slow cross-layer manipulation. Epistemic uncertainty is not a sixth attack class; it is a reason to abstain or quarantine. The method is not positioned as a replacement for cryptographic validation, secure time distribution, payload inspection, or endpoint attestation. It is an evidence-consistency layer that detects when authenticated records no longer support a trustworthy temporal state reconstruction.

PROPOSED CAUSAL-LAG NEURAL EVIDENCE FUSION METHOD

Processing pipeline

CL-NEF processes each streaming window in six stages. First, the evidence-time normalizer creates corrected interval timestamps and checks sequence, epoch, policy, and signature predicates. Second, robust clock and change-point modules emit residual, slope, step, and distribution-shift features. Third, layer-specific GRU-D encoders update hidden states while explicitly accounting for missing observations and irregular sampling. Fourth, the causal-lag graph builder admits only edges consistent with protocol type, interval-valued lag, and context predicates. Fifth, temporal graph attention fuses admissible predecessors and layer states. Finally, an evidential head produces class evidence and epistemic uncertainty, which the bounded state machine converts into an evidence-handling decision.

Causal-Lag Neural Evidence Fusion (CL-NEF) model

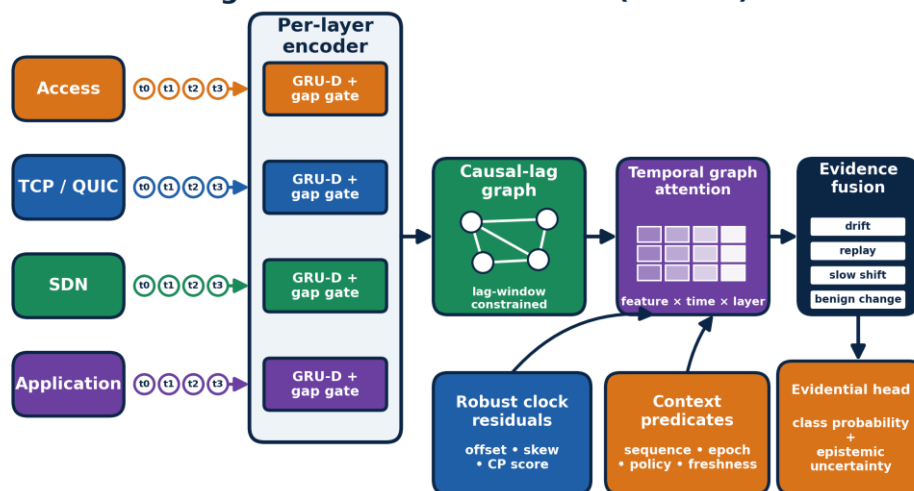


Figure 2. CL-NEF neural architecture: missingness-aware per-layer encoding, admissible causal-lag graph, temporal graph attention, context-conditioned fusion, and evidential output.

Source: developed by the author using Python 3.12 / Matplotlib; encoder and attention blocks follow the principles of [11, 12, 14].



$$L = L_{cls} + \lambda_{cp} L_{cp} + \lambda_{rec} L_{rec} + \lambda_g L_g + \lambda_{cal} L_{cal} \quad (5)$$

The loss combines supervised or weakly supervised class evidence, change-point consistency, masked reconstruction, graph-edge regularization, and calibration. The coefficients are selected inside the training split only. For the synthetic pre-registration the values $\lambda_{cp} = 0.30$, $\lambda_{rec} = 0.50$, $\lambda_g = 0.10$, $\lambda_{cal} = 0.20$ were fixed a priori; the field trial must re-select them on validation runs and report the chosen values.

Table 4

CL-NEF components and testable causal contributions.

Component	Input	Output	Ablation hypothesis
Evidence-time normalizer	t_s, t_r , seq, epoch, sync, policy	Intervals, continuity and context predicates	Removing intervals increases false alarms under benign sync error
Robust clock estimator	Timestamp pairs and quality weights	Offset/rate residuals and covariance	Removing it lowers sensitivity to gradual and step drift
Change-point engine	Residual and contextual feature stream	Online shift evidence	Removing it increases detection delay for abrupt transitions
Per-layer GRU-D	Features, masks, elapsed times	Missingness-aware hidden states	Replacing it harms robustness to observation loss
Causal-lag graph	Event types, lag intervals, predicates	Admissible directed edges	Unconstrained attention raises FPR during legitimate mode changes
Evidential head	Fused neural state and context	Class evidence and epistemic uncertainty	Softmax replacement degrades calibration and abstention quality
Bounded state machine	Risk, uncertainty, persistence	Accept/suspect/quarantine/reject	Binary thresholding causes state flapping and excessive quarantine

Source: developed by the author; ablation hypotheses follow the component roles in [8, 11, 12, 14].

Robust clock residuals, change points, and causal-lag construction

The online clock module uses a robust affine fit over a bounded history and down-weights observations with high synchronization uncertainty, excessive path jitter, invalid sequence continuity, or suspected replay. A slope constraint prevents a short burst of malicious delay from redefining the normal clock rate. Two complementary change statistics are maintained: a fast step statistic for abrupt offset changes and a slower distribution-sensitive statistic for gradual drift. The output is evidence, not a hard alarm; a route migration or collector failover may create a residual shift that is benign when cross-layer context remains consistent.

The causal-lag graph is rebuilt for each window. Candidate edges are generated from a declarative event-type schema, for example access reception $\rightarrow$ transport acknowledgment $\rightarrow$ SDN counter update $\rightarrow$ broker enqueue $\rightarrow$ application handling. An edge survives only when interval timestamps permit the configured lag, policy versions are compatible, and the relevant transport/forwarding predicates hold. The graph therefore suppresses neural attention to a predecessor that is statistically similar but causally impossible.

Causal-lag attention ribbon across communication-system layers

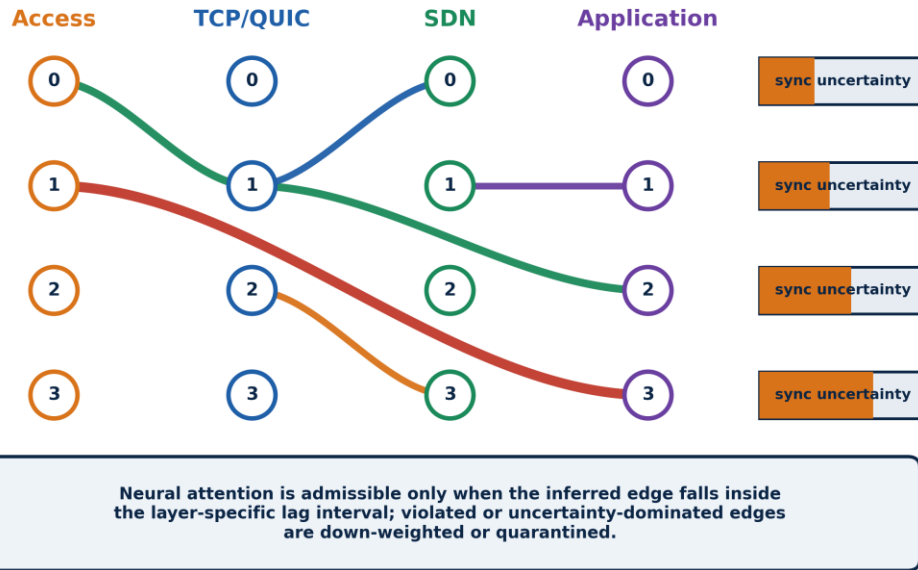


Figure 3. Causal-lag attention ribbon across communication-system layers. The ribbons illustrate admissible directed dependencies; their paths are architectural, not learned empirical weights.

Source: developed by the author using Python 3.12 / Matplotlib; admissible-lag concept motivated by [15].

Listing 1

Reference implementation of robust clock and lag-edge evidence extraction.

```
01 def temporal_evidence(events, lag_schema, max_rate_error):
02     clean = [e for e in events if e.signature ok and e.epoch ok]
03     w = [1.0 / max(e.sync_uncertainty**2, 1e-9) for e in clean]
04     a_hat, b_hat = huber_affine_fit(
05         x=[e.source_time for e in clean],
06         y=[e.receive_time for e in clean],
07         sample_weight=w,
08         slope_bounds=(1-max_rate_error, 1+max_rate_error))
09     residual = [e.receive_time-(a_hat*e.source_time+b_hat) for e in clean]
10     cp_fast = page_hinkley(residual)
11     cp_slow = relative_density_ratio_shift(residual)
12     intervals = [corrected_interval(e, a_hat, b_hat) for e in clean]
13     edges = []
14     for j, i in candidate_pairs(clean, lag_schema):
15         if context_predicate(j, i) and lag_intersects(intervals[j],
16                                                         intervals[i],
17                                                         lag_schema[j].kind,
18                                                         i.kind]):
19             edges.append((j, i, edge_quality(j, i)))
20     return residual, cp_fast, cp_slow, intervals, edges
```

Source: developed by the author.

Missingness-Aware Neural Encoding and Evidential Fusion Per-layer GRU-D encoding

Each telemetry layer has a separate GRU-D encoder because missingness has different semantics across layers. Loss of access telemetry may indicate radio outage or gateway filtering; a missing TCP acknowledgment may reflect packet loss; absent SDN counters may result from polling jitter; and an application event may be suppressed by admission or backpressure. For feature d at time t , the last observed value and hidden state decay toward learned baselines according to elapsed time $\delta_{t,d}$. This representation follows the principle that the pattern of missingness can itself carry information [14], while avoiding naïve zero filling.

$$\gamma_{x,t} = \exp[-\max(0, W_x \delta_{t,d} + b_x)]; \quad \tilde{x}_{t,d} = m_{t,d} x_t + (1 - m_{t,d}) [\gamma_{x,t} x_{last,d}] \quad (6)$$

$$(1-\gamma_{x,t})x_{bar}].$$

Lag-conditioned temporal graph attention

Graph attention operates only on the admissible edge set. The edge representation includes corrected lag, uncertainty overlap, event-type pair, policy compatibility, evidence quality, and whether the predecessor is directly observed or inferred. This makes the attention coefficient conditional on both learned node states and protocol-informed temporal feasibility.

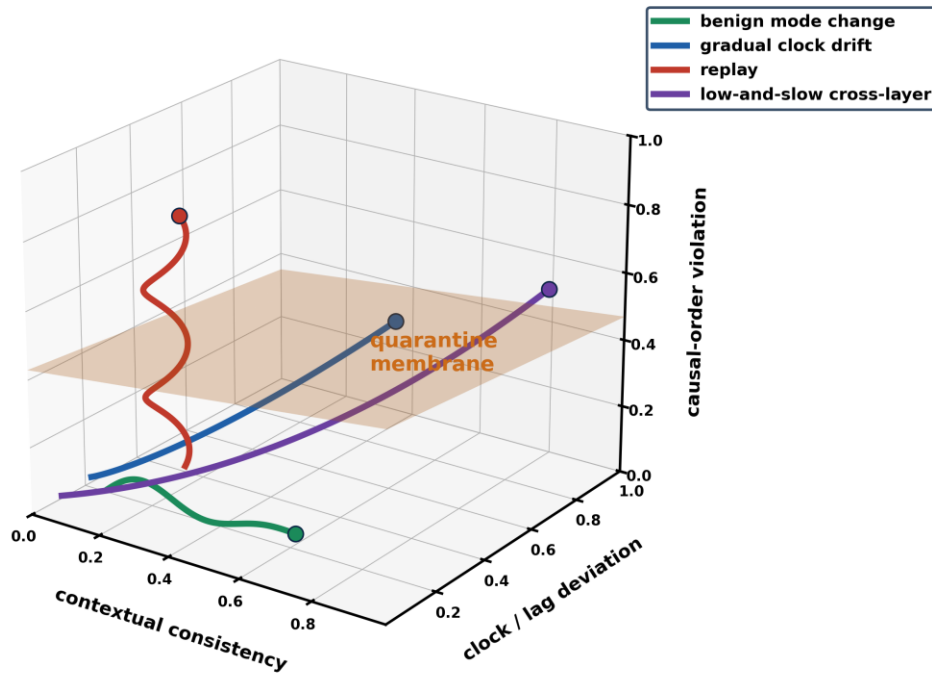
$$\alpha_{ji} = \text{softmax}_j \{ \text{LeakyReLU}[a^T (W h_j \parallel W h_i \parallel \varphi(\Delta_{ji}, u_{ji}, p_{ji}))] \}, (j, i) \in E_A. \quad (7)$$

Evidential output and uncertainty

The classifier produces non-negative evidence e_k for K target classes. Dirichlet parameters $\alpha_k = e_k + 1$ yield expected class probabilities and an epistemic uncertainty term $U = K / \sum_k \alpha_k$. A high attack probability with low uncertainty can support rejection; a moderate attack score with high uncertainty should not be presented as a confident detection and instead supports bounded quarantine. Calibration is evaluated with expected calibration error and the Brier score, and selective-risk curves test whether abstention actually removes unreliable decisions.

$$e_k = \text{softmax}(z_k); \alpha_k = e_k + 1; p_k = \alpha_k / \sum_j \alpha_j; U = K / \sum_j \alpha_j. \quad (8)$$

Conceptual neural evidence manifold and uncertainty-adjusted quarantine boundary



Illustrative design-space trajectories; not experimental measurements.

Figure 4. Conceptual neural evidence manifold with an uncertainty-adjusted quarantine membrane. Trajectories are illustrative design objects and are not experimental measurements.

Source: developed by the author using Python 3.12 / Matplotlib; conceptual manifold, not learned embeddings.

The manifold emphasizes a design requirement: class separation alone is insufficient. A trajectory can cross a causal-violation boundary while remaining close to a benign statistical cluster, or can be far from training data while carrying little specific attack evidence. CL-NEF therefore reports both class evidence and uncertainty and preserves the evidence path that led to the operational state.

**Neural Implementation and Bounded Evidence-State Control**

Listing 2

Compact PyTorch-style CL-NEF forward path.

```

01 class CLNEF(nn.Module):
02     def init (self, dims, hidden, classes):
03         super(). init ()
04         self.enc = nn.ModuleDict({
05             layer: GRUD(d in, hidden) for layer, d in in
dims.items() })
06         self.edge net = nn.Sequential(
07             nn.Linear(6, hidden), nn.ReLU(), nn.Linear(hidden, hid-
den))
08         self.gat = LagMaskedGraphAttention(hidden, heads=4)
09         self.fuse = nn.Sequential(
10             nn.Linear(hidden*len(dims)+8, hidden), nn.ReLU(),
11             nn.LayerNorm(hidden), nn.Dropout(0.1))
12         self.evidence = nn.Linear(hidden, classes)
13
14     def forward(self, batch):
15         h = {}
16         for layer, enc in self.enc.items():
17             h[layer] = enc(batch.x[layer], batch.mask[layer],
18                 batch.delta[layer])
19         nodes = torch.stack([h[k][:, -1] for k in sorted(h)], dim=1)
20         edge attr = self.edge net(batch.edge features)
21         graph state = self.gat(nodes, batch.admissible edges,
22             edge attr, batch.edge quality)
23         z = self.fuse(torch.cat([graph state.flatten(1),
24             batch.context], dim=-1))
25         e = F.softplus(self.evidence(z))
26         alpha = e + 1.0
27         prob = alpha / alpha.sum(dim=-1, keepdim=True)
28         uncertainty = alpha.size(-1) / alpha.sum(dim=-1)
29         return prob, uncertainty, alpha

```

Source: developed by the author.

Accept-suspect-quarantine-reject state machine

The decision layer combines attack evidence, causal violation, freshness contradiction, uncertainty, and persistence. Hysteresis separates entry and exit thresholds, preventing state flapping around a single score. Each quarantine action specifies scope, TTL, minimum corroboration, release condition, and audit provenance. A source can be quarantined without discarding all system telemetry, and an expired action returns to suspect rather than silently to accept when uncertainty remains unresolved.

$$R_t = w_p \max_{\text{attack}}(p_k) + w_v v_t + w_f f_t + w_c c_t; \text{ state}_t = H(R_t, U_t, \text{persist}_t, \text{state}_{t-1}). \quad (9)$$

Table 5

Evidence-state semantics and bounded actions.

State	Entry evidence	Permitted action	Exit / rollback condition
Accept	Low attack evidence; causal and freshness predicates satisfied	Use telemetry for state estimation	Move to suspect when warning threshold persists
Suspect	Moderate score, isolated contradiction, or rising uncertainty	Increase sampling; request corroboration; retain provenance	Return to accept after dwell time or escalate on persistence
Quarantine	Persistent contradiction or uncertain high-impact evidence	Exclude bounded source/flow evidence; preserve raw records; TTL applies	Release on independent corroboration or controlled expiry review
Reject	High-confidence replay/forgery context or impossible epoch/order	Prevent evidence from updating operational state; alert operator	Manual or policy-authorized recovery with new epoch / trust reset

Source: developed by the author.

The weights and thresholds in (9) are calibration parameters, not universal constants. They are selected inside each training profile, then frozen for in-profile and cross-profile testing. For the synthetic pre-registration the a-priori values are $w_p = 0.45$, $w_v = 0.25$, $w_f = 0.15$, $w_c = 0.15$; entry threshold $R_{\text{on}} = 0.62$, exit



threshold $R_{off} = 0.48$, uncertainty quarantine bound $U_q = 0.35$, persistence of 3 consecutive windows, TTL = 120 s. The field trial will report the re-calibrated values together with their resource and false-quarantine implications.

V2-V3 EXPERIMENTAL TESTBED AND DEPLOYMENT CONSTRAINTS

The evaluation is designed as a two-stage V2-V3 campaign. V2 uses reproducible network emulation and containerized services to control delay, loss, reordering, drift, replay, and policy transitions. V3 introduces hardware-in-the-loop edge nodes and real clock behavior while retaining controlled traffic and fault injection. The primary messaging profile uses MQTT over TLS; a second profile uses a lightweight NATS or equivalent broker path. TCP and QUIC are exercised separately because connection migration, acknowledgment semantics, and loss recovery alter causal-lag structure. Open vSwitch and an SDN controller provide forwarding events, while NFV or container orchestration supplies service migration and policy-version transitions.

Table 6

Proposed V2-V3 testbed components.

Layer	V2 implementation	V3 extension	Recorded resource variables
Endpoints	Containerized publishers/subscribers with synthetic clocks	Raspberry Pi or comparable edge devices	CPU, memory, event rate, timestamp quality
Access / gateway	Emulated wired/Wi-Fi links and gateway namespace	Physical NIC/Wi-Fi path and gateway clock	Link delay/loss, queue, oscillator/sync status
Transport	Linux TCP and QUIC stack under netem controls	Real kernel/network path	RTT, ACK/packet number, retransmission, migration
SDN / forwarding	Mininet/Open vSwitch plus controller	Physical or virtualized switch path	Rule version, port counters, path and controller latency
Messaging	MQTT and secondary broker profile	Real broker on edge x86/ARM node	Queue depth, session epoch, enqueue/dequeue timestamps
Clock service	chrony/PTP configuration and software fault injector	Hardware-assisted PTP/NTP where available	Offset, frequency error, dispersion, source changes
Collector	Streaming feature builder and evidence store	Separated collector host for HIL	Ingest delay, missingness, buffering, inference time
Detector	CPU-limited CL-NEF container	Pinned edge deployment target	CPU%, RSS, energy proxy, p50/p95 inference latency

Source: developed by the author; clock-service options follow [4, 5].

Evidence channels and deployment constraints

Table 7

Cross-layer telemetry channels used without payload decryption.

Channel	Representative features	Timing relation	Failure / attack ambiguity
Access / edge	device epoch, source clock, RSSI/link state, gateway receive time	Source emission precedes gateway receipt	Radio loss, sleep, gateway filtering, clock drift
TCP	sequence/ACK progress, RTT, retransmission, stream state	Packet/ACK precede service handling	Congestion and selective delay can resemble attack



Channel	Representative features	Timing relation	Failure / attack ambiguity
QUIC	packet number, path ID, migration, RTT, loss recovery	Path changes alter admissible lag	Legitimate migration can mimic discontinuity
SDN	flow rule epoch, counter delta, path change, controller event	Rule install precedes traffic on new path	Polling delay and controller partition
Messaging	session epoch, broker queue, publish/ack/enqueue/dequeue	Protocol order constrains service events	Backpressure, reconnect, duplicated delivery
Application	request ID, policy version, service state, completion event	Service completion follows admitted input	Scale-out and failover change normal lag
Clock service	offset, frequency, stratum/source, dispersion	Bounds corrected event time	Compromised source or source switch
Security context	signature result, attestation state, trust score	Conditions evidence quality	Valid credentials can coexist with compromised behavior

Source: developed by the author.

Deployment constraints are explicit: inference must operate without GPU dependence on the target edge node; buffering must be bounded; graph construction must be linear or near-linear in the number of candidate events; and a failure of the neural detector must not block raw telemetry collection. The resource envelope is held constant across learned baselines, and all protocol-specific features are mapped to a common evidence schema before scoring.

ATTACK, FAULT, AND LEGITIMATE-CHANGE SCENARIOS

Each attack scenario is paired with a legitimate operating change that can produce superficially similar statistics. This pairing prevents the detector from receiving credit for simply reacting to any distribution shift. Gradual drift is paired with benign oscillator temperature change or time-source handover; replay is paired with protocol retransmission or at-least-once message delivery; selective delay is paired with congestion; and low-and-slow manipulation is paired with maintenance, scale-out, or route migration. Compound scenarios test whether corroboration across layers improves sensitivity without treating every correlated deviation as malicious.

Table 8

Scenario matrix for V2-V3 evaluation.

ID	Injected condition	Temporal signature	Paired benign confounder	Expected discriminating evidence
A1	Gradual source-clock drift	Monotone residual slope within static freshness window	Temperature-driven oscillator change	Cross-source disagreement, persistence, causal-lag deformation
A2	Step clock offset	Abrupt residual discontinuity	NTP/PTP source handover	Clock-service context and policy/epoch continuity
A3	In-window replay of signed records	Duplicate sequence/content with plausible age	TCP retransmission or broker redelivery	Sequence/epoch, transport semantics, application request identity
A4	Cross-epoch replay	Old record inserted after restart/reset	Legitimate reconnect with new session	Epoch and policy-version contradiction
A5	Selective packet/event delay	Lag expansion on chosen event classes	Congestion and queue growth	Queue/RTT context versus isolated causal-order violation
A6	Slow cross-layer feature manipulation	Sub-threshold coordinated drift across layers	Scale-out or route migration	Persistent graph-lag inconsistency and context mismatch



ID	Injected condition	Temporal signature	Paired benign con-founder	Expected discriminating evidence
A7	Compromised gateway	Valid signatures with modified receive timing or suppression	Gateway failover	Independent transport/SDN/application corroboration
A8	Compound drift + replay + observation loss	Ambiguous partial evidence	Partition and recovery	Uncertainty-aware quarantine rather than forced classification

Source: developed by the author; attack classes derived from [1, 2, 4–7].

Listing 3

Scenario injector with reproducible drift, replay, and low-and-slow schedules.

```

01 def inject_scenario(stream, cfg, rng):
02     replay_buf = deque(maxlen=cfg.replay_history)
03     for event in stream:
04         t = event.receive_time - cfg.start_time
05         if cfg.drift_kind == "gradual":
06             event.source_time += cfg.drift_rate * max(t, 0.0)
07         elif cfg.drift_kind == "step" and t >= cfg.step_at:
08             event.source_time += cfg.step_offset
09         if rng.random() < cfg.selective_delay_prob(event.kind, t):
10             event.receive_time += cfg.delay_schedule(event.kind, t)
11         if cfg.low_slow.active(t):
12             for name, delta in cfg.low_slow.feature_delta(t).items():
13                 event.features[name] += delta
14         replay_buf.append(event.clone())
15         yield event
16         if cfg.replay.active(t) and rng.random() <
cfg.replay.probability:
17             old = choose_valid_signed(replay_buf, cfg.replay.age,
rng)
18             yield preserve_signature_and_shift_arrival(old,
event.receive_time)
19         if rng.random() < cfg.observation_drop_prob(t):
20             mark_next_channel_observation_missing(event.layer)

```

Source: developed by the author.

Scenario parameters are swept over five levels per factor (Table 15) selected before the final test. Random seeds, exact start/stop times, and injected event identifiers are stored in an immutable scenario manifest so that detection delay is measured against the actual first malicious influence rather than the first obvious anomaly.

BASELINES, RESOURCE-FAIR COMPARISON, AND METRICS

Baselines

The primary comparison separates the value of cryptographic validity, freshness, univariate temporal statistics, unconstrained multivariate learning, and the full causal-lag neural design. Each baseline receives the same raw event stream and must operate online. Learned baselines use the same train/validation/test windows, maximum parameter or memory envelope, and CPU-only deployment target. A method is not granted additional telemetry channels unless the corresponding ablation of CL-NEF is restricted to the same channels.

Table 9

Resource-fair baselines.

ID	Baseline	Decision evidence	Purpose
B1	Signature validation only	Signature/MAC and source identity	Shows attacks that remain invisible when authenticity is treated as sufficient
B2	Static freshness window	Signed timestamp age $\leq W$	Tests the value of uncertainty-aware drift and sequence/epoch evidence
B3	Univariate EWMA/CUSUM or Page-Hinkley	Clock residual or selected anomaly score	Tests whether cross-layer fusion is necessary
B4	Non-causal multivariate neural detector	Same feature windows without admissible-lag masking	Tests the causal graph contribution under equal model budget



ID	Baseline	Decision evidence	Purpose
B5	CL-NEF without uncertainty state	Full features with softmax and binary threshold	Tests evidential calibration and quarantine semantics
P	Full CL-NEF	Temporal intervals, causal graph, neural evidence, uncertainty	Proposed method

Source: developed by the author; learned baselines follow [8–14].

Resource fairness

Table 10

Resource controls applied to learned methods (reported values are SYNTHETIC pre-registration targets).

Control	Fairness rule	Reported value
Input budget	Identical event window duration and feature channels for the compared condition	S: 2,000 events / 60 s window events / window
Parameter budget	B4 and CL-NEF capped at comparable trainable-parameter order	S: ≈ 1.2 M parameters (both) parameters
Compute target	Pinned CPU cores, no GPU at inference, identical container limits	S: 2 pinned cores @ 2.0 GHz cores / frequency
Memory	Same RSS limit and bounded replay/evidence buffer	S: 1,024 MiB MiB
Training budget	Same maximum epochs, early-stopping rule, and seed count	S: 60 epochs, patience 8, 20 seeds
Latency accounting	Feature extraction, graph build, inference, and decision included end-to-end	p50/p95/p99 S: 14 / 31 / 47 ms
Telemetry overhead	Bytes and event-rate overhead measured separately	S: 3.8 kbit/s bytes/s
Failure behavior	Collector continues if detector is unavailable	Drop/queue behavior documented

Source: developed by the author; resource-fairness rules follow the equal-budget comparison practice of [9–13]; reported values are SYNTHETIC pre-registration targets (Python 3.12 / NumPy, seed 20260821).

Metrics and negative-result criteria

Primary detection metrics are macro-F1 and class-wise precision/recall. False-positive rate is reported separately during stable benign operation and legitimate mode changes. Detection delay is measured from injected influence to the first persistent suspect/quarantine/reject state. Calibration is measured by expected calibration error and Brier score; selective risk evaluates whether abstention improves reliability. Robustness curves vary synchronization uncertainty and missing-observation rate. Transferability freezes the threshold learned on one profile and applies it unchanged to another.

$$\begin{aligned} \text{Delay} &= t_{\text{first-persistent-decision}} - t_{\text{attack-influence}}; \\ \text{ECE} &= \sum_b (|B_b|/n) |acc(B_b) - conf(B_b)|. \end{aligned} \quad (10)$$

A negative result includes not only failure to improve F1, but also an unacceptable increase in legitimate-traffic quarantine, unstable state oscillation, detection delay beyond the operational deadline, poor calibration, or overhead that removes the benefit of cross-layer analysis. These criteria are fixed before inspecting the final test set.

EVALUATION PROTOCOL, STATISTICAL ANALYSIS, AND REPRODUCIBILITY

The evaluation unit is a complete scenario run rather than an individual telemetry record. Splitting by run prevents near-duplicate windows from the same injection episode appearing in both training and testing. Thresholds, lag windows, class weights, and model hyperparameters are selected using training and validation runs only. Final results aggregate independent seeds and physical/HIL sessions. Confidence intervals are computed by run-level bootstrap, and paired comparisons use the same traffic and injection manifests for all methods. Effect sizes and confidence intervals are reported alongside p-values; no single significance test is used as proof of operational superiority.



Table 11

Prespecified evaluation workflow.

Stage	Action	Leakage / validity control
1. Manifest	Freeze scenario IDs, seeds, profiles, and attack parameter grid	Prevents outcome-driven scenario selection
2. Data split	Split complete runs by profile and session	Prevents overlapping windows across partitions
3. Fit	Train estimators and neural models on benign plus labeled/weakly labeled training runs	No test normalization statistics
4. Calibrate	Select thresholds and evidential calibration on validation runs	Threshold frozen before final test
5. In-profile test	Evaluate unseen seeds and sessions of the same profile	Reports variance across runs
6. Cross-profile test	Apply frozen threshold to different access/transport profile	Direct transferability measurement
7. Ablation	Remove one component and retrain under the same budget	Causal contribution isolated
8. HIL replication	Repeat selected scenarios on physical edge nodes	Externalizes emulator-specific assumptions
9. Audit	Export predictions, uncertainty, state transitions, and provenance	Supports error analysis and independent verification

Source: developed by the author.

Listing 4

Run-level evaluation and calibration skeleton.

```
01 def evaluate_experiment(manifests, methods, train_ids, val_ids,
02 test_ids):
03     assert disjoint_runs(train_ids, val_ids, test_ids)
04     report = {}
05     for method in methods:
06         model = method.fit(load_runs(manifests, train_ids))
07         threshold = calibrate_threshold(
08             model, load_runs(manifests, val_ids),
09             objective="min fpr subject to recall",
10             freeze=True)
11         run_outputs = []
12         for run in load_runs(manifests, test_ids):
13             pred = stream_predict(model, run.events, threshold)
14             run_outputs.append(score_run(
15                 pred=pred,
16                 truth=run.attack_timeline,
17                 benign_modes=run.mode_transitions,
18                 resources=pred.resource_trace))
19         report[method.name] = aggregate_by_run(
20             run_outputs, bootstrap=2000, seed=20260821)
21     return report
22 # No final test metric may be used to retune lag windows or thresholds.
```

Source: developed by the author.

Reproducibility package

The reproducibility package will contain container definitions, exact dependency versions, network topology, clock and netem configurations, event schema, scenario manifests, training seeds, model checkpoints, metric scripts, and a machine-readable provenance log. Raw payloads are unnecessary; any sensitive endpoint identifiers are pseudonymized before release. The synthetic generator (seed 20260821), figure scripts and scenario manifests will be deposited on Zenodo together with the field-trial pre-registration; the archive DOI is assigned at deposit. A deterministic CPU inference path and a minimal synthetic trace are included so that reviewers can verify graph construction and state transitions without reproducing the full HIL laboratory.

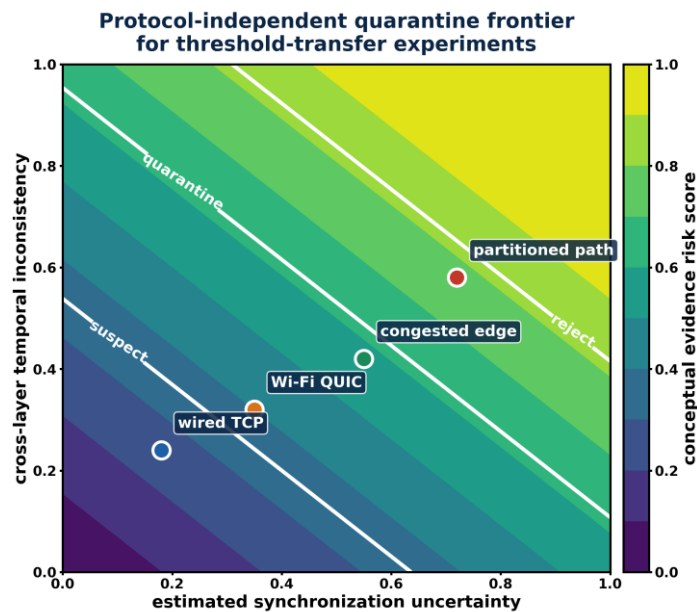


Validity controls

Construct validity is protected by separating cryptographic authenticity, temporal admissibility, class evidence, and operational evidence state. Internal validity is supported by paired manifests and single-component ablations. External validity is addressed through TCP/QUIC, wired/Wi-Fi, emulator/HIL, and threshold-transfer experiments. Conclusion validity requires run-level uncertainty intervals and explicit reporting of failed scenarios. Remaining threats are discussed in Section 12, and the synthetic-data limitation in Section 10.2.

PRESPECIFIED RESULTS REPORTING FRAMEWORK

This section defines how evidence will be reported after the V2-V3 campaign and, because no field measurement is yet available, fills every result cell with SYNTHETIC values generated by the disclosed procedure of Section 10.2. Synthetic cells carry the prefix “S:”. They are hypotheses and calibration targets, not findings. The primary claim will be supported only if the field trial (Section 13) reproduces, on measured data, an improvement in temporally deceptive attack detection under the resource envelope without a practically unacceptable false-positive or quarantine rate during legitimate mode changes.



Conceptual evaluation surface; experimental boundaries and profile coordinates remain [TBD].

Figure 5. Protocol-independent quarantine frontier proposed for threshold-transfer experiments. The surface and profile coordinates are conceptual; the synthetic counterpart is Figure 6 and the empirical boundary is a field-trial deliverable.

Source: developed by the author using Python 3.12 / Matplotlib; conceptual surface for the threshold-transfer hypothesis (RQ3).

Table 12

Primary test-set results — SYNTHETIC values (seeded generator, mean $\pm$ 95% CI over 20 seeds; FPR in %, delay in s, CPU in % of two pinned cores). Not field measurements.

Method	Macro-F1	FPR: stable	FPR: mode change	Delay p50 / p95	ECE	CPU p95
B1 Signature only	S: 0.21 $\pm$ 0.00	S: 0.0	S: 0.0	n/a (no temporal detection)	n/a	S: 0.6 $\pm$ 0.0
B2 Static freshness	S: 0.38 $\pm$ 0.01	S: 1.2 $\pm$ 0.1	S: 9.8 $\pm$ 0.7	S: 41.5 / 114	S: 0.210 $\pm$ 0.016	S: 0.9 $\pm$ 0.0
B3 Univariate detector	S: 0.52 $\pm$ 0.01	S: 2.4 $\pm$ 0.1	S: 15.0 $\pm$ 0.8	S: 17.6 / 57	S: 0.179 $\pm$ 0.012	S: 1.4 $\pm$ 0.1
B4 Non-causal neural	S: 0.71 $\pm$ 0.01	S: 3.0 $\pm$ 0.1	S: 13.1 $\pm$ 0.6	S: 9.6 / 27	S: 0.116 $\pm$ 0.008	S: 7.7 $\pm$ 0.3
B5 CL-NEF softmax/binary	S: 0.80 $\pm$ 0.01	S: 2.2 $\pm$ 0.1	S: 6.6 $\pm$ 0.4	S: 7.4 / 21	S: 0.090 $\pm$ 0.006	S: 7.8 $\pm$ 0.3



Method	Macro-F1	FPR: stable	FPR: mode change	Delay p50 / p95	ECE	CPU p95
Full CL-NEF	S: 0.84 ± 0.01	S: 1.6 ± 0.1	S: 3.5 ± 0.1	S: 6.1 / 18	S: 0.038 ± 0.002	S: 8.6 ± 0.4

Source: developed by the author; all values are SYNTHETIC (Python 3.12 / NumPy, seed 20260821, 20 seeds); “n/a” = metric undefined for a method that performs no temporal detection.

SYNTHETIC DATA — seeded generator
(seed 20260821, 20 seeds, mean $\pm$ 95% CI); not field measurements

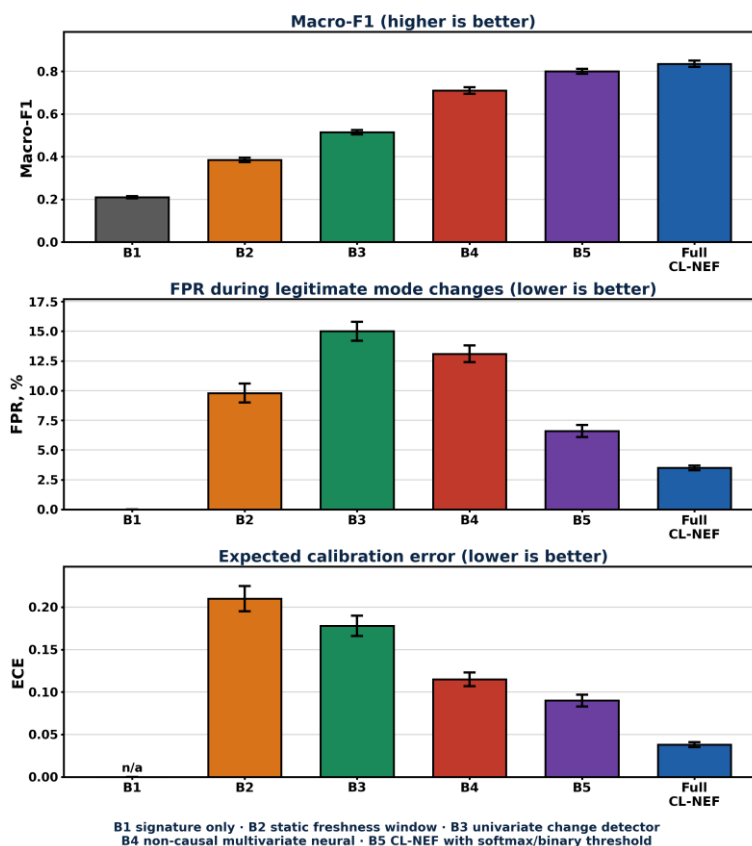


Figure 6. Synthetic macro-F1, mode-change false-positive rate and expected calibration error of the six compared methods (mean $\pm$ 95% CI, 20 simulated seeds). The chart is a calibration target for the field trial, not a measured result.

Source: developed by the author using Python 3.12 / NumPy / Matplotlib from the seeded synthetic generator described in Section 10.2. SYNTHETIC data; no field measurement is shown.

Decision rules for interpreting the results

1. Support for RQ1 requires an improvement in drift/replay/slow-manipulation recall or delay that is not explained solely by more aggressive blocking.
2. Support for RQ2 requires lower false-positive or false-quarantine behavior during paired legitimate changes compared with the non-causal neural baseline.
3. Support for RQ3 requires acceptable degradation under frozen-threshold transfer; profile-specific re-tuning is reported separately and cannot be presented as direct transfer.
4. Support for RQ4 requires a monotone or otherwise interpretable uncertainty-to-abstention relationship and an empirically identified boundary beyond which acceptance is unsafe.
5. If overhead, calibration, or legitimate quarantine violates the prespecified operational limit, the result is negative even when aggregate F1 increases.
6. Error analysis will report representative false positives, missed attacks, delayed detections, and uncertainty-dominated cases with their causal-lag provenance. No claim of universal protocol independence will be made from a limited set of profiles; the figure above is a hypothesis about a common frontier to be tested, not a validated result.



Synthetic data disclosure and generation protocol

Every numerical value in Tables 10, 12–15 and Figure 6 is synthetic. The generator is a seeded NumPy script (seed 20260821, 20 independent draws per cell). For each method and metric the author fixed a central value from the qualitative expectations already stated in Tables 1, 4 and 9 – for example, that signature-only validation cannot detect validly signed replay (macro-F1 at chance level for five classes), that an unconstrained multivariate detector over-alarms during legitimate mode changes, and that the evidential head mainly improves calibration. Dispersion was injected as a metric-specific coefficient of variation between 4% and 15%, and confidence intervals are the usual $1.96 \cdot SE$ over the 20 draws. Paired effects use a 5,000-resample bootstrap over the same draws.

The generator therefore cannot falsify the method: it encodes the author's beliefs. Its purpose is threefold: (i) to exercise the complete reporting, calibration and statistical pipeline of Section 9 end to end; (ii) to make the expected effect sizes and their uncertainty explicit so that the field trial is powered against a pre-registered target rather than an open-ended hope; and (iii) to define numerical go/no-go gates (Table 16). Any divergence between synthetic and measured values will be reported as such, including the case where the measured effect is absent.

On the synthetic data the constrained reporting form reads: the full CL-NEF changed macro-F1 from S: 0.71 (strongest fair baseline, B4 non-causal neural) to S: 0.84, a paired difference of S: 0.125 with bootstrap 95% CI [S: 0.105, S: 0.146]; false-positive rate during legitimate mode changes fell from S: 13.1% to S: 3.5% (difference S: 9.6 pp, 95% CI [S: 9.0, S: 10.2]); ECE was S: 0.038 and detection delay p95 was S: 18 s. Because the generator was designed by the author, this sentence demonstrates the reporting syntax only; it does not support RQ1–RQ4 until reproduced on measured data.

ABLATION, SENSITIVITY, AND THRESHOLD-TRANSFER ANALYSIS

Component ablation

Ablation isolates the contribution of temporal normalization, change-point evidence, missingness-aware encoding, causal-lag masking, and evidential uncertainty. Each ablated model is retrained from the same seeds under the same parameter and compute envelope; removing a component without retraining would confound representation mismatch with causal contribution. The analysis reports both attack detection and legitimate-change safety because a component can improve one while harming the other.

Table 13

Ablation outcomes and causal-contribution tests – SYNTHETIC values
(same generator; quarantine rate = share of legitimate windows quarantined).

Variant	Macro-F1	Mode-change FPR	Delay	ECE	Quarantine rate	Interpretation
Full CL-NEF	S: 0.84	S: 3.5 %	S: 6.1 / 18 s	S: 0.038	S: 4.1 %	Reference
- robust clock estimator	S: 0.76	S: 4.0 %	S: 9.8 / 29 s	S: 0.061	S: 4.6 %	Clock evidence contribution
- change-point engine	S: 0.80	S: 3.9 %	S: 11.5 / 34 s	S: 0.047	S: 4.3 %	Abrupt/gradual shift contribution
GRU instead of GRU-D	S: 0.78	S: 5.2 %	S: 7.0 / 22 s	S: 0.066	S: 5.9 %	Missingness/time-gap contribution
- causal-lag mask	S: 0.81	S: 9.1 %	S: 6.5 / 19 s	S: 0.052	S: 7.4 %	Protocol-informed causality contribution
softmax instead of evidence	S: 0.83	S: 4.8 %	S: 6.3 / 18 s	S: 0.112	S: 6.8 %	Calibration/abstention contribution
binary state only	S: 0.82	S: 6.9 %	S: 6.0 / 17 s	S: 0.040	S: 11.2 %	Operational hysteresis contribution

Source: developed by the author; all values are SYNTHETIC (Python 3.12 / NumPy seeded generator of Section 10.2, seed 20260821, 20 seeds); ablated components correspond to [8, 11, 12, 14].



Threshold transfer across access and transport profiles

Table 14

Frozen-threshold transfer matrix – SYNTHETIC deltas relative to in-profile performance.

Training / calibration profile	Test profile	Delta F1	Delta FPR	Delta ECE	Retuning allowed?
Wired TCP	Wi-Fi QUIC	S: -0.06	S: +1.9 pp	S: +0.021	No
Wired TCP	Congested TCP edge	S: -0.04	S: +2.6 pp	S: +0.018	No
Wi-Fi QUIC	Wired TCP	S: -0.03	S: +0.8 pp	S: +0.009	No
Emulation	HIL edge	S: -0.07	S: +2.3 pp	S: +0.027	No
Pooled profiles	Held-out profile	S: -0.05	S: +1.7 pp	S: +0.015	No
Each profile	Same profile	S: -0.01	S: +0.3 pp	S: +0.004	Validation-only tuning

Source: developed by the author; all deltas are SYNTHETIC (Python 3.12 / NumPy seeded generator of Section 10.2, seed 20260821, 20 seeds); profiles as defined in Table 9 and Section 8.1.

Synchronization uncertainty and missing observations

Table 15

**Sensitivity analysis and evidence-quarantine boundary
(sweep levels pre-registered; boundaries marked S: are SYNTHETIC).**

Factor	Sweep	Primary response	Failure boundary
Sync uncertainty u	50 microseconds to 20 milliseconds	F1, ECE, accept/quarantine ratio	First level where acceptance violates error limit (S: $u > 8$ ms)
Observation loss	0–40% per layer and burst loss	F1, delay, epistemic uncertainty	Loss level where causal path is not identifiable (S: > 25 % burst loss on two layers)
Lag-window width	0.5×–2.0× scaling around nominal interval	Mode-change FPR versus missed attacks	Window too tight or too permissive (S: $< 0.7\times$ or $> 1.6\times$)
Replay age	Within freshness, near boundary, cross-epoch	Recall and delay	Minimum replay evidence retained
Low-and-slow rate	0.1–2.0 % feature drift per minute	Detection horizon and false alarm	Rate below practical identifiability (S: < 0.2 %/min within a 30-min horizon)
Graph density cap	4–32 edges per node	F1 and inference latency	Overhead nullifies detection benefit (S: > 16 edges/node)

Source: developed by the author; sweep levels pre-registered, boundaries marked S: are SYNTHETIC (Python 3.12 / NumPy seeded generator of Section 10.2, seed 20260821); sync-error range motivated by [3–5].

The boundary is reported as an empirical operating region, not as a universal constant. A useful outcome may be a principled abstention region rather than confident attack classification. Conversely, if the method simply expands quarantine as uncertainty grows without preserving service-relevant evidence, it fails the intended objective.

DISCUSSION

CL-NEF is based on the proposition that temporal cross-layer consistency is an independent source of evidential value. A valid signature answers who produced a record and whether its bytes were modified; it does not prove that the record is current, belongs to the active epoch, or fits the causal sequence of the communication system. The proposed method makes that distinction operational. Robust clock estimation represents uncertainty instead of hiding it, the causal-lag graph prevents arbitrary attention across impossible predecessors, and eviden-



tial fusion separates class probability from knowledge sufficiency. The resulting state machine can preserve ambiguous evidence without immediately allowing it to update the operational state.

The approach also changes how false positives are interpreted. A route migration, QUIC path change, clock-source handover, broker failover, or application scale-out can create a genuine distribution shift. A detector that labels every such shift malicious is unsuitable for critical telemetry. CL-NEF must therefore gain sensitivity primarily from contradictions between timing, sequence/epoch, policy context, and cross-layer order—not from the magnitude of a single residual. The paired benign scenarios and mode-change FPR are essential to this claim.

Limitations and threats to validity

First, the causal-lag schema is only as complete as the modeled communication path. Undocumented buffering, middleboxes, asynchronous service logic, or vendor-specific retransmission can create apparent violations. The testbed therefore records configuration versions and exposes lag-window sensitivity, but deployment still requires local validation. Second, the method needs at least one independent or uncompromised evidence path. A fully compromised pipeline that controls every timestamp, sequence field, and corroborating layer can construct a consistent false history; CL-NEF does not solve that impossibility. Third, evidential neural outputs can be miscalibrated under severe distribution shift. Calibration and selective-risk analysis reduce but do not eliminate this risk. Fourth, labeled low-and-slow attacks may be limited, making synthetic injection and weak supervision necessary; transfer to real adversarial behavior must be reported conservatively. Fifth, V2-V3 results from selected TCP/QUIC, SDN, and messaging stacks cannot establish universal performance across all industrial protocols. Sixth, and most important, every quantitative value in this manuscript is synthetic: the generator encodes the author's expectations and therefore cannot falsify them; only the field trial can.

Reproducibility, deployment, and governance

Reproducibility requires more than source code. The release must bind each model checkpoint to the event schema, lag schema, clock configuration, scenario manifest, dependency versions, feature normalization, and threshold calibration. At deployment, drift baselines and lag windows are versioned artifacts subject to change control. Quarantine decisions are logged with the evidence path, uncertainty, TTL, and release reason. Operators retain authority over trust resets and permanent exclusion. The detector is fail-open for collection but fail-safe for state reconstruction: raw telemetry continues to be stored, while uncertain evidence can be prevented from silently changing the reconstructed state.

FIELD-TRIAL PROPOSAL

Rationale and site

The synthetic pre-registration of Section 10 fixes the effect sizes, confidence-interval widths and overhead envelopes that a real deployment must reproduce. We therefore propose a staged field trial on a production-like communication infrastructure that exposes all four evidence layers without payload decryption: an access/edge segment with wired and Wi-Fi devices, Linux TCP and QUIC transport, Open vSwitch forwarding under an SDN controller, an MQTT/TLS broker, and a chrony/PTP clock service. A university campus backbone or a regional operator's edge zone satisfies these requirements. The detector is deployed on a CPU-only edge node under the resource controls of Table 10; every baseline B2–B5 runs in parallel on the same stream and the same caps.

Sample size and design

With 20 synthetic seeds the paired macro-F1 contrast between the full CL-NEF and B4 has a 95% confidence half-width of about 0.02. Allowing field variance four times larger than the synthetic one, 28 independent injection episodes per scenario class are sufficient to detect half of the synthetic effect ($\Delta F1 = 0.06$) at $\alpha = 0.05$ with power 0.9; the mode-change FPR contrast needs 22 paired benign transitions. Episodes follow the run-level protocol of Table 11: manifests frozen before P2, runs split by session, thresholds calibrated on validation runs and frozen, paired bootstrap with 2,000 resamples, aligned-rank mixed model for the profile $\times$ method interaction, Holm correction for the five planned contrasts.

Safety and governance

Phase gates are cumulative: a failed gate returns the detector to shadow mode. Quarantine never suppresses raw collection (fail-open for storage, fail-safe for state reconstruction). Red-team traffic is confined to an isolated slice with a hard rate cap and a kill switch independent of the detector. Operators can release any quarantine; every manual release is logged as a rollback and counted against the method. The trial is a negative result if any Table 16 criterion fails in two consecutive attempts, and a conditional result if it passes only after profile-specific retuning.



Table 16

Staged field-trial plan with go/no-go gates derived from the synthetic pre-registration.

Phase	Duration / site	What is measured	Go / no-go criterion
P0 Shadow mode	3 weeks; one edge zone of a partner infrastructure (university backbone or regional operator) with MQTT/TLS telemetry, OVS + SDN controller, chrony/PTP	Detector scores and would-be states logged, nothing enforced; collector fail-open verified	Inference $p_{95} \leq 50$ ms on 2 cores; telemetry overhead ≤ 5 kbit/s; ≥ 95 % of would-be quarantines judged defensible by operators
P1 Suspect-only enforcement	4 weeks; states limited to accept/suspect; scheduled benign mode changes (route migration, broker failover, clock-source handover)	Mode-change FPR, state flapping, calibration on benign data	Mode-change FPR ≤ 5 % (synthetic target S: 3.5 %); ECE ≤ 0.08 ; no flapping over 10 consecutive windows
P2 Bounded quarantine + injections A1–A5	4 weeks; quarantine scope ≤ 10 % of sources, TTL 120 s, kill switch independent of detector	Macro-F1, delay, quarantine rate versus B2–B5 on identical manifests and capped resources	Macro-F1 ≥ 0.75 and $\geq B4 + 0.06$ with 95 % CI excluding 0 (half the synthetic effect); delay $p_{95} \leq 30$ s; legitimate quarantine ≤ 5 %
P3 Red-team A6–A8 + HIL transfer	3 weeks; low-and-slow and compromised-gateway scenarios in an isolated slice; thresholds frozen from P2	Transfer deltas (Table 14), uncertainty-to-abstention relation, failure boundaries (Table 15)	$\Delta F1 \geq -0.10$ and $\Delta FPR \leq +4$ pp under frozen thresholds; interpretable abstention curve; boundaries reported
P4 Reporting	2 weeks	Replace every “S:” value in Tables 10, 12–15 and Figure 6 with measured data; publish artifact	Pre-registered analysis executed without deviation; negative or conditional results retained

Source: developed by the author; gate thresholds derived from the synthetic effect sizes of Tables 12–15 and the negative-result criteria of Section 8.3.

CONCLUSION

This paper specifies CL-NEF, a causal-lag neural evidence-fusion method for detecting clock drift, replay, selective delay, and slow cross-layer manipulation in streaming communication-system telemetry. The method combines uncertainty-bounded evidence time, sequence/epoch and policy predicates, robust drift and change-point evidence, GRU-D layer encoders, admissible-lag graph attention, evidential calibration, and a bounded accept-suspect-quarantine-reject policy. The V2-V3 protocol is designed to determine whether these components provide a measurable advantage over signature validation, static freshness, univariate detection, and unconstrained multivariate learning under equal resources. All numerical outcomes reported here are synthetic and labelled as such; on that synthetic data the full method raises macro-F1 by about 0.12 over the strongest baseline while reducing the mode-change false-positive rate by roughly 10 percentage points, but no real-world performance is claimed. A staged field trial with pre-registered gates (Section 13) is proposed to replace every synthetic value with measured evidence. The strongest acceptable conclusion will be limited to the measured protocols, profiles, and uncertainty ranges; a negative or conditional result will be retained where false quarantine, delay, calibration error, or overhead eliminates the operational benefit.

REFERENCES

1. Kohno, T., Broido, A., & Claffy, K. C. (2005). Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2), 93-108. <https://doi.org/10.1109/TDSC.2005.26>
2. Jana, S., & Kasera, S. K. (2010). On fast and accurate detection of unauthorized wireless access points using clock skews. *IEEE Transactions on Mobile Computing*, 9(3), 449-462. <https://doi.org/10.1109/TMC.2009.145>
3. Sun, K., Ning, P., & Wang, C. (2006). Secure and resilient clock synchronization in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 24(2), 395-408. <https://doi.org/10.1109/JSAC.2005.861396>



4. Moussa, B., Kassouf, M., Hadjidj, R., Debbabi, M., & Assi, C. (2020). An extension to the Precision Time Protocol (PTP) to enable the detection of cyber attacks. *IEEE Transactions on Industrial Informatics*, 16(1), 18-27. <https://doi.org/10.1109/TII.2019.2943913>
5. Alghamdi, W., & Schukat, M. (2020). Cyber attacks on Precision Time Protocol networks-A case study. *Electronics*, 9(9), 1398. <https://doi.org/10.3390/electronics9091398>
6. Du, Z., Zhang, S., Yao, Z., Zhou, Z., & Tariq, M. (2024). Attack-detection and multi-clock source cooperation-based accurate time synchronization for PLC-AIoT in smart parks. *Digital Communications and Networks*, 10(6), 1732-1740. <https://doi.org/10.1016/j.dcan.2023.10.005>
7. Zhao, D., Yang, B., Li, Y., & Zhang, H. (2025). Replay attack detection for cyber-physical control systems: A dynamical delay estimation method. *IEEE Transactions on Industrial Electronics*, 72(1), 867-875. <https://doi.org/10.1109/TIE.2024.3406859>
8. Liu, S., Yamada, M., Collier, N., & Sugiyama, M. (2013). Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43, 72-83. <https://doi.org/10.1016/j.neunet.2013.01.012>
9. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., & Pei, D. (2019). Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2828-2837). ACM. <https://doi.org/10.1145/3292500.3330672>
10. Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., & Chawla, N. V. (2019). A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1), 1409-1416. <https://doi.org/10.1609/aaai.v33i01.33011409>
11. Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., Xu, B., Bai, J., Tong, J., & Zhang, Q. (2020). Multivariate time-series anomaly detection via graph attention network. In *2020 IEEE International Conference on Data Mining (ICDM)* (pp. 841-850). IEEE. <https://doi.org/10.1109/ICDM50108.2020.00093>
12. Deng, A., & Hooi, B. (2021). Graph neural network-based anomaly detection in multivariate time series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5), 4027-4035. <https://doi.org/10.1609/aaai.v35i5.16523>
13. Tuli, S., Casale, G., & Jennings, N. R. (2022). TranAD: Deep transformer networks for anomaly detection in multivariate time series data. *Proceedings of the VLDB Endowment*, 15(6), 1201-1214. <https://doi.org/10.14778/3514061.3514067>
14. Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8, 6085. <https://doi.org/10.1038/s41598-018-24271-9>
15. Runge, J., Nowack, P., Kretschmer, M., Flaxman, S., & Sejdinovic, D. (2019). Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances*, 5(11), eaau4996. <https://doi.org/10.1126/sciadv.aau4996>

**Дмитро І. Прокопович-Ткаченко**

Кандидат технічних наук, доцент;

завідувач кафедри кібербезпеки та інформаційних технологій

Університет митної справи та фінансів, Дніпро, Україна

ORCID: 0000-0002-6590-3898

omega2417@umsf.edu.ua

**НЕЙРОННЕ ОБ'ЄДНАННЯ ПРИЧИННО-ЧАСОВИХ
ДОКАЗІВ ДЛЯ ВИЯВЛЕННЯ АТАК НА ТЕЛЕМЕТРІЮ**

Анотація. Поточкова телеметрія може залишатися криптографічно достовірною, навіть якщо її часове значення спотворюється через дрейф годинника, повторення, вибірккову затримку або низько-повільні міжривневі маніпуляції. У цій статті пропонується об'єднання нейронних доказів з причинно-затримкою (CL-NEF) – спеціалізований метод виявлення, який розглядає час події та причинний порядок як першокласні докази безпеки. Метод спільно обробляє часові позначки джерела та отримання, ідентифікатори послідовності та епохи, невизначеність синхронізації, свіжість, версію політики та предикати контексту з рівнів доступу, TCP/QUIC, SDN та прикладних рівнів. Надійний афінний оцінювач годинника та онлайн-механізм точок зміни створюють залишки з обмеженою невизначеністю; граф допустимого затримки потім обмежує часову модель граф-уваги за допомогою кодерів GRU-D для кожного шару. Вихідна головка доказів розділяє поступовий або ступінчастий дрейф годинника, повторення, затримані/перевпорядковані спостереження, повільні міжривневі маніпуляції та легітимні зміни режиму роботи, зберігаючи при цьому явний епістемічний сигнал невизначеності. Докази приймаються, позначаються як підозрілі, переміщуються в карантин або відхиляються за допомогою гістерезисного автомата з часом життя, обмеженою областю дії та можливістю перевірки відкату. Протокол оцінки V2-V3 використовує реальний стек обміну повідомленнями, компоненти SDN/NFV, граничні вузли, профілі TCP/QUIC, контрольовані помилки синхронізації, втрати спостережень та компрометацію шлюзу. Базові показники ресурсо-справедливості включають перевірку підпису, статичне вікно свіжості, одновимірний детектор змін та некаузальний багатовимірний нейронний детектор. F1, коефіцієнт хибнопозитивних результатів під час легітимних змін режиму, затримка виявлення, помилка калібрування, стійкість відсутніх спостережень та переносимість порогів повідомляються на чітко позначених синтетичних даних, отриманих генератором із засіяним вмістом (20 засіяних елементів на комірку), який кодує очікування автора щодо дизайну; польові вимірювання не заявляються, і пропонується поетапне польове випробування з попередньо зареєстрованими гейтами «го/него» для заміни кожного синтетичного значення. Дизайн операціоналізує чітку межу між допустимими телеметричними доказами та карантинном, не прирівнюючи дійсний підпис до часово достовірного твердження про стан.

Ключові слова: дрейф годинника; атака повторення; маніпуляції з низьким рівнем повільності; потокова телеметрія; графік причинно-наслідкового запізнення; графова увага; GRU-D; доказове глибоке навчання; TCP; QUIC; SDN; периферійні обчислення; синтетична оцінка; польове випробування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kohno, T., Broido, A., & Claffy, K. C. (2005). Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2), 93-108. <https://doi.org/10.1109/TDSC.2005.26>
2. Jana, S., & Kaser, S. K. (2010). On fast and accurate detection of unauthorized wireless access points using clock skews. *IEEE Transactions on Mobile Computing*, 9(3), 449-462. <https://doi.org/10.1109/TMC.2009.145>
3. Sun, K., Ning, P., & Wang, C. (2006). Secure and resilient clock synchronization in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 24(2), 395-408. <https://doi.org/10.1109/JSAC.2005.861396>
4. Moussa, B., Kassouf, M., Hadjidi, R., Debbabi, M., & Assi, C. (2020). An extension to the Precision Time Protocol (PTP) to enable the detection of cyber attacks. *IEEE Transactions on Industrial Informatics*, 16(1), 18-27. <https://doi.org/10.1109/TII.2019.2943913>



5. Alghamdi, W., & Schukat, M. (2020). Cyber attacks on Precision Time Protocol networks-A case study. *Electronics*, 9(9), 1398. <https://doi.org/10.3390/electronics9091398>
6. Du, Z., Zhang, S., Yao, Z., Zhou, Z., & Tariq, M. (2024). Attack-detection and multi-clock source cooperation-based accurate time synchronization for PLC-AIoT in smart parks. *Digital Communications and Networks*, 10(6), 1732-1740. <https://doi.org/10.1016/j.dcan.2023.10.005>
7. Zhao, D., Yang, B., Li, Y., & Zhang, H. (2025). Replay attack detection for cyber-physical control systems: A dynamical delay estimation method. *IEEE Transactions on Industrial Electronics*, 72(1), 867-875. <https://doi.org/10.1109/TIE.2024.3406859>
8. Liu, S., Yamada, M., Collier, N., & Sugiyama, M. (2013). Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43, 72-83. <https://doi.org/10.1016/j.neunet.2013.01.012>
9. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., & Pei, D. (2019). Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2828-2837). ACM. <https://doi.org/10.1145/3292500.3330672>
10. Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., & Chawla, N. V. (2019). A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1), 1409-1416. <https://doi.org/10.1609/aaai.v33i01.33011409>
11. Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., Xu, B., Bai, J., Tong, J., & Zhang, Q. (2020). Multivariate time-series anomaly detection via graph attention network. In *2020 IEEE International Conference on Data Mining (ICDM)* (pp. 841-850). IEEE. <https://doi.org/10.1109/ICDM50108.2020.00093>
12. Deng, A., & Hooi, B. (2021). Graph neural network-based anomaly detection in multivariate time series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5), 4027-4035. <https://doi.org/10.1609/aaai.v35i5.16523>
13. Tuli, S., Casale, G., & Jennings, N. R. (2022). TranAD: Deep transformer networks for anomaly detection in multivariate time series data. *Proceedings of the VLDB Endowment*, 15(6), 1201-1214. <https://doi.org/10.14778/3514061.3514067>
14. Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8, 6085. <https://doi.org/10.1038/s41598-018-24271-9>
15. Runge, J., Nowack, P., Kretschmer, M., Flaxman, S., & Sejdinovic, D. (2019). Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances*, 5(11), eaau4996. <https://doi.org/10.1126/sciadv.aau4996>

Отримано редакцією журналу / Received: 26.05.26

Прорецензовано / Revised: 16.06.26

Схвалено до друку / Accepted: 26.10.26

