Smart Technologies Academic Press

Jordanian Journal of Informatics and Computing
https://www.jjic.thestap.com/



Secure-Aggregation Federated Graph Attention Networks with SHAP-Based Edge Attribution for Advanced Persistent Threat Detection

Ahmed Fakhri Abdullah[1*], Ahmed Sedeeq Baker Al-Doori[2]

[1]*Ministry of Higher Education and Scientific Research, 10011, Baghdad, Iraq*
[2]*Northern Technical University / College of Health and Medical Techniques - Al-Dour*

ARTICLE INFO

Article History

Received: 07-05-2026
Revised: 30-06-2026
Accepted: 08-11-2026

Vol.2027, No.1

DOI:

***Corresponding author.**

Email
alpasha.online@yahoo.com

Orcid:

This is an open access article under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

Published by STAP Publisher.



ABSTRACT

Advanced Persistent Threats (APTs) pose significant cybersecurity challenges due to their stealthy, prolonged activities and the limited visibility of malicious behaviour in host provenance data. Existing detection approaches often require centralized access to sensitive logs and provide limited interpretability, raising concerns about data confidentiality and analyst trust. To address these challenges, this study proposes a privacy-preserving and explainable federated learning framework for APT detection using provenance graphs. The proposed framework integrates edge-typed Graph Attention Networks (GATv2), self-supervised masked-feature learning, and federated prototype-based anomaly detection to identify suspicious activities without requiring labeled attack data during training. To preserve confidentiality, a dropout-resilient, Bonawitz-style secure aggregation protocol is employed to protect model updates and shared statistics, including normalization parameters, clustering information, and calibration distributions. Furthermore, edge-level Shapley value attribution is incorporated to provide interpretable explanations of detected anomalies and support security analysts during incident investigation. The framework is evaluated using the DARPA Transparent Computing Engagement 3 benchmark, specifically the THEIA and CADETS datasets, under a simulated four-client federated setting. Experimental results across three random seeds demonstrate strong detection performance, achieving Area Under the Receiver Operating Characteristic Curve (AUROC) values of 0.995 and 0.971 and Area Under the Precision–Recall Curve (AUPRC) values of 0.870 and 0.561 for THEIA and CADETS, respectively. The proposed secure aggregation mechanism maintains detection performance comparable to conventional federated averaging, introduces approximately 0.1% additional training time on THEIA, and reduces cross-round client-linkage attack accuracy from 85.6% to chance-level performance. Behaviour-level analysis demonstrates complete detection of scan-like attack activities at a calibrated 95% threshold, while other malicious activities exhibit threshold-dependent recall. The explainability analysis confirms the completeness of Shapley attributions and convergence of the highest-ranked contributing edges within 25 permutations, although their empirical fidelity is comparable to less computationally expensive explanation methods. Overall, the findings demonstrate the potential of combining secure federated learning, provenance-graph representation learning, and explainable AI for confidentiality-preserving APT detection, while highlighting important limitations related to threshold sensitivity, explanation fidelity, and semi-honest security assumptions.

Keywords: Advanced Persistent Threats (APTs); Federated Learning; Provenance Graphs; Graph Attention Networks (GATv2); Secure Aggregation; Anomaly Detection; Cybersecurity.

How to cite the article

1. Introduction

1.1 Background and Motivation

Advanced Persistent Threats (APTs) are long-running, multi-stage intrusion campaigns in which a well-resourced adversary gains a foothold, escalates privileges, moves laterally, and exfiltrates data over weeks or months while deliberately keeping each individual action indistinguishable from legitimate activity **[1]**. Signature-based and volume-based detectors are structurally ill-suited to this "low-and-slow" behaviour, because no single event is malicious in isolation; what is anomalous is the *causal context* in which events occur. Host audit data (Linux audited, FreeBSD/DTrace, Windows ETW) can be transformed into a *provenance graph*, in which processes, files, sockets and other kernel objects are nodes and the system calls that connect them are typed, time-stamped, directed edges. Fig. 1 illustrates the setting.

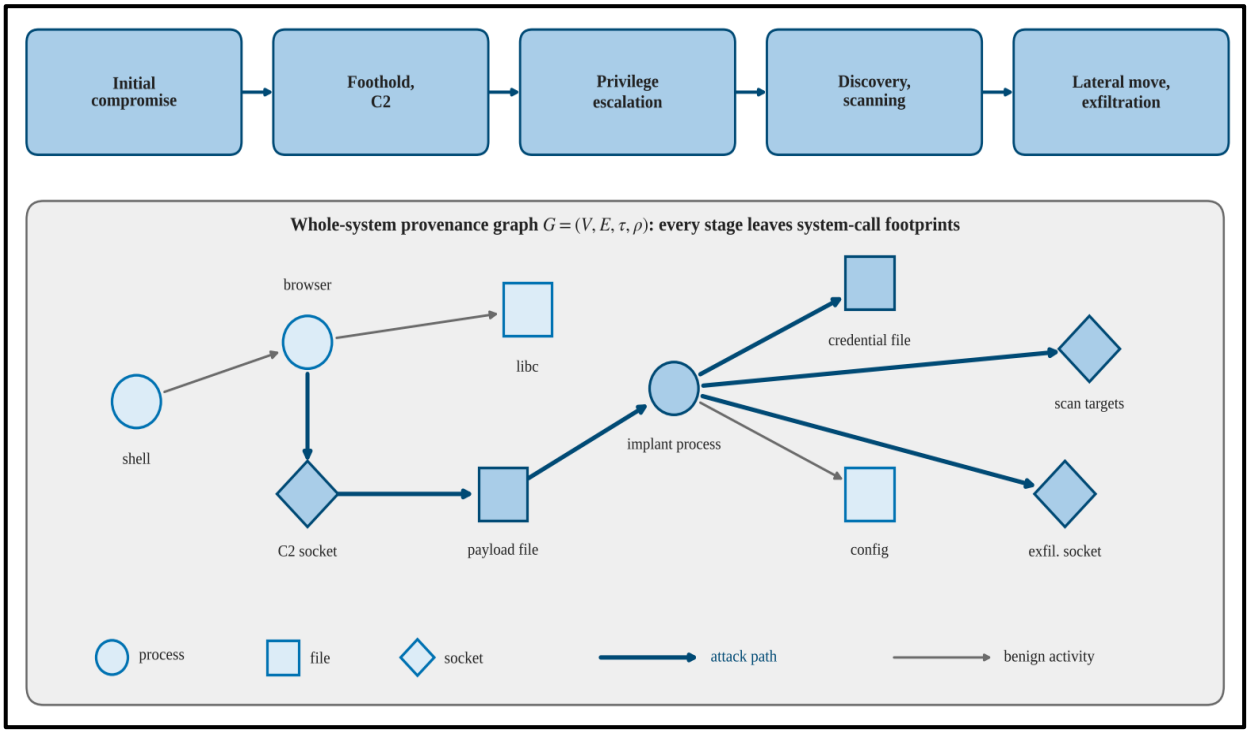


Figure 1. An APT campaign (top) leaves a trace in the whole-system provenance graph (bottom): the attack path (red) is a small subgraph of mostly benign activity (grey), which is why node-level detection on the graph is the unit of analysis.

A first generation of provenance-based detectors relied on hand-written rules and tag propagation over this graph **[2]**. A second generation replaced rules with learning: graph sketches and clustering for whole-graph anomalies **[3]**, path- and sequence-based models **[4]**, node-level GraphSAGE models **[5]**, recommendation-style embeddings of entity interactions **[6]**, temporal graph networks **[7]**, semantic-aware node embeddings **[8]**, online fine-grained attack subgraph discovery **[9]**, and masked graph autoencoders built on graph attention **[10]**. The community's de-facto benchmark is the DARPA Transparent Computing (TC) Engagement 3 release **[11]**, on which several of these systems report near-perfect detection. A careful recent re-examination nevertheless shows that this headline performance is hard to translate into deployment, and that a simple baseline can match much more complex architectures on several of the datasets **[12]**. Two structural properties of the field are rarely questioned in that literature, and both are the starting point of this paper.

First, training is centralised. Every system cited above assumes that raw or lightly processed audit logs are shipped to a single analytics server. Audit logs, however, are among the most sensitive telemetry an organisation produces: they contain executable paths, command lines, user names, internal host names, IP addresses and file names that may encode customer or personal data. Managed security providers, multi-tenant clouds and threat-sharing consortia therefore cannot simply pool them. Federated learning (FL) **[13]** is the natural alternative: each data owner trains locally and only model updates travel. But model updates are not private. Gradients and weight deltas can be inverted to recover training inputs **[14],** and for a graph model trained on provenance data the recoverable information includes degree and edge-type statistics that characterise exactly which services a host runs. *Secure aggregation* (SecAgg) **[15]** closes this gap by ensuring that the server learns only the sum of the client updates, never an individual one.

Second, detections are opaque. A graph neural network emits a score for a node; a security-operations analyst must decide, under time pressure, whether to isolate a machine. Without a statement of *which parts of the neighborhood* caused the score, the alert is expensive to triage. Post-hoc attribution methods such as SHAP **[16]** address this for tabular models, and graph-specific explainers exist **[17]**, but they have not, to our knowledge, been evaluated on provenance graphs against analyst ground truth, nor on a model trained in a federated, secure-aggregation setting.

This paper addresses both gaps with a single coherent system: a **Secure-Aggregation Federated Graph Attention Network (FedGAT)** for APT detection on provenance graphs, with **SHAP-based edge attribution** that explains every flagged node in terms of the provenance edges that drive its anomaly score. Every component named in the title is implemented and evaluated, not simulated: the secure aggregation layer uses real X25519 key agreement, HKDF, AES-256-GCM, a ChaCha20-based PRG and Shamir secret sharing, and the experiments use the publicly released DARPA TC Engagement 3 provenance graphs.

1.2 Problem Statement

Let K data owners ("clients") each hold a private provenance graph G_k of benign activity. A semi-honest aggregation server coordinates training. At deployment time a graph G^{test} that may contain an APT campaign is scored. We want to solve three coupled problems.

- **P1 (detection).** Learn, without any attack labels and without any client revealing its graph or its individual update, a scoring function $s: V \to \mathbb{R}_{\geq 0}$ that ranks the entities of G^{test} taking part in the attack above benign entities. How close can federated, securely-aggregated training come to centralised training, what does each design choice (edge-type awareness, masked-feature pre-text task, attention) contribute, and *which kinds of attack behaviour* (scanning, exploitation, command-and-control, implant) does the detector actually find?
- **P2 (secure aggregation).** Make every cross-client quantity, not only the model update, leak only an aggregate: this includes normalisation statistics, clustering statistics and score histograms that the detector needs. What does this cost in exactness, bandwidth and computation, and does it tolerate client dropout?
- **P3 (explanation).** For a flagged node v, assign to each provenance edge in its neighbourhood a signed contribution ϕ_e such that the contributions provably sum to the node's anomaly score. Are these contributions *faithful* to the model, *stable* under Monte-Carlo estimation, and *aligned* with the entities that analysts have labelled as malicious?

1.3 Contributions

- **A fully federated, label-free APT detector**: an edge-type-aware GATv2 encoder trained with a masked-feature objective and scored by distance to per-node-type prototypes, with every cross-client statistic (normalisation, clustering, threshold calibration) expressed as a *sum* so that the whole pipeline runs under secure aggregation.
- **A real, dropout-tolerant secure aggregation layer** on standard primitives, with an exactness bound, measured overhead, and an empirical test that the server-side view of a client contribution is indistinguishable from uniform.
- **An exactly-efficient edge-level Shapley explainer** using permutation sampling with an *interventional* value function in which a removed edge is replaced by a benign-typical message.
- **An empirical study on DARPA TC E3** (THEIA and CADETS) with centralised, federated, plain-versus-secure, local-only and raw-feature baselines, ablations, multiple seeds, and an audit that excludes.
- **Threat-type-level detection analysis**: the labelled attack of each host is decomposed into entity classes, related to the published scenarios and MITRE ATT&CK techniques, with detection and alert workload per class.
- **A candid evaluation of explanation quality** against random, attention, saliency, GNNExplainer-style and occlusion baselines, including negative findings.

1.4 Organization of the Paper

Section 2 reviews related work, Section 3 states the threat model, Section 4 the dataset and its audit, Section 5 the methodology, Section 6 the experimental protocol, Section 7 the results, Section 8 the discussion and limitations, and Section 9 concludes.

2. Related Work

2.1 Provenance-Based APT Detection

Backtracking intrusions through dependency graphs of OS events dates back to the 2000s. SLEUTH **[18]** and HOLMES **[19]** propagate tags through the graph and match behaviours to adversary tactics; they are fast and interpretable but depend on expert-written policies. UNICORN **[20]** summarises the evolving graph into a fixed-size sketch and detects anomalous windows with clustering, giving a whole-graph, unsupervised verdict. ProvDetector **[21]** and ATLAS **[22]** learn from rare

causal paths and event sequences. threaTrace **[23]** trains one GraphSAGE model per node type on benign data to detect individual malicious *nodes*, which is the granularity we adopt. ShadeWatcher **[24]** casts entity interactions as a recommendation problem. Kairos **[25]** uses a temporal graph network for edge-level reconstruction and then constructs an attack summary; FLASH **[26]** adds semantic embeddings of node attributes; NodLink **[27]** formulates online attack-subgraph discovery as a Steiner-tree problem; and MAGIC **[28]** pre-trains a graph attention autoencoder with a masked-feature objective and detects outliers by distance to benign embeddings, which is the closest precursor to our encoder. Surveys and critical studies organise this literature **[29],** and document that evaluation on the TC datasets is sensitive to label construction and to train/test overlap **[30]**, a theme we revisit in §4.5. Two lines of work probe robustness: mimicry attacks that graft benign behaviour onto malicious actions **[31]** and adversarial system actions that evade ML detectors **[32]**. None of the above is federated, none runs under secure aggregation, and none attributes a node's score to individual provenance edges with a game-theoretic guarantee.

2.2 Graph Neural Networks and Graph Anomaly Detection

Message-passing networks underlie GCN **[33]**, GraphSAGE **[34]** and relational variants such as R-GCN **[35]**, which give every edge type its own transformation. Graph attention **[36]** weights neighbours by a learned, input-dependent coefficient; GATv2 **[37]** fixes the static-attention limitation of the original formulation by applying the non-linearity before the attention vector. Provenance edges carry crucial type information (read, write, exec, connect,), so our encoder injects a learned edge-type embedding into both the message and the attention logit rather than treating the graph as homogeneous. For unsupervised anomaly detection, deep one-class objectives and attributed-network auto encoders score a node by reconstruction error or distance to a learned normal region; masked graph auto encoding **[38]** provides a stronger pretext task than plain reconstruction by hiding a node's own attributes and requiring them to be recovered from its context. Our scoring function is deliberately a *distance to benign prototypes* in embedding space, a choice driven as much by federated compatibility as by accuracy.

2.3 Federated Learning and Federated Graph Learning

FedAvg **[15]** alternates local stochastic optimization with averaging of the resulting models; FedProx **[39]** adds a proximal term to limit client drift under heterogeneity; adaptive server optimisers have also been proposed. The open problems of the field are surveyed by Kairouz et al. **[16]**. Federated graph learning is younger: FedGraphNN **[40]** provides a benchmark, FedSage+ **[41]** recovers missing cross-client neighbours, and FedPerGNN **[42]** personalizes recommendation GNNs while hiding user-item interactions. Federated intrusion detection has been explored for IoT traffic **[43]** and surveyed more broadly, but on flow features or sequences rather than provenance graphs, and generally without cryptographic aggregation. Our setting differs in that each client's graph is a *causal* structure whose node degrees and edge-type mixtures are themselves sensitive, and in that the detector needs cross-client *statistics* (not only gradients) that we also protect **[44].**

2.4 Secure Aggregation and Its Limits

Bonawitz et al. **[19]** construct a protocol in which each pair of clients derives a shared pseudorandom mask from a Diffie-Hellman key agreement; masks cancel in the sum, while Shamir secret sharing **[44]** lets surviving clients help the server remove the masks of dropped clients. Bell et al. **[20]** reduce the overhead to polylogarithmic by replacing the complete communication graph with a sparse random graph, and Prio provides a complementary approach for private aggregate statistics with input validation. Why aggregation matters is shown by gradient inversion, which reconstructs inputs from a single client's update **[17], [18]**. SecAgg is not a panacea: a malicious server can send different models to different clients and thereby isolate an individual's update **[27]**, and the aggregate itself still leaks, which is the motivation for adding differential privacy **[28]**.

2.5 Gap Analysis

Table 1 positions this work against representative systems along the dimensions that matter for the problem stated in §1.2. Entries describe each system's primary mechanism as published and are not a performance ranking.

Table 1. Representative provenance and federated detection systems versus this work.

System	Learning	Granularity	Federated	Secure agg.	Edge-level attribution
HOLMES **[3]**	rules / TTP correlation	alerts, kill-chain	no	no	tactic mapping
UNICORN **[4]**	graph sketch + clustering	whole graph	no	no	no
threaTrace **[7]**	GraphSAGE per type	node	no	no	no
ShadeWatcher **[8]**	recommendation-style GNN	interaction	no	no	no
Kairos **[9]**	temporal GNN	edge / summary	no	no	no
FLASH **[10]**	GNN + semantic embeddings	node	no	no	no
NodLink **[11]**	online, Steiner tree	attack subgraph	no	no	no
MAGIC **[12]**	masked GAT autoencoder	node / graph	no	no	no
DÏoT **[43]**	federated GRU (IoT traffic)	device traffic	yes	no	no
This work	**edge-typed GATv2, masked AE**	**node**	**yes**	**yes (all statistics)**	**yes (Shapley)**

3. Preliminaries and Threat Model

3.1 Provenance Graph

A provenance graph is a typed directed multigraph $G = (V, E, \tau, \rho)$ with node-type map $\tau: V \to \{1, \dots, K_n\}$ and edge-type map $\rho: E \to \{1, \dots, K_e\}$. Each edge $e = (u, v, r)$ records that subject or object u acted on v through an event of type r. Consistent with the preprocessing of, events whose semantics are data *inflow* (read, receive, load) are stored in the direction of information flow, so that an edge always points from the information source to its sink. For message passing we add, for every edge (u, v, r), a reverse edge $(v, u, r + K_e)$, so the encoder sees $2K_e$ relation types and every node can aggregate from both its causal parents and its causal children. The in-neighbourhood of v in the augmented graph is

$$\mathcal{N}(v) = \{(u, r): (u, v, r) \in E\} \cup \{(u, r + K_e): (v, u, r) \in E\}. \quad (1)$$

3.2 Threat Model

We consider K clients (data owners) and one aggregation server. The server is semi-honest: it follows the protocol but tries to learn individual client contributions from everything it receives. Up to $K - t$ clients may drop out in any round, where $t = \lfloor K/2 \rfloor + 1$ is the Shamir threshold; the server and at most $t - 1$ clients may collude without learning an honest client's input. Clients are honest in the sense that they do not poison their updates. The *assets* we protect are (i) every per-round model updateΔ_k, (ii) the client's feature moments, (iii) its clustering statistics and (iv) its score histogram; the *assets we do not protect* are the final model, the final prototypes, and the aggregate histograms, all of which are, by design, available to every participant. The explainer runs at the data owner (or analyst) on the owner's own graph; only the (public) model and prototypes are needed, so no raw graph leaves its owner during explanation either. Table 2 summarizes the parties and the protections. Notation is introduced where used: $d = K_n + 2K_e + 2$ is the feature dimension, $h = 64$ the hidden width, $H = 4$ heads of width $C = 16$, K the number of clients, t the Shamir threshold, $s(v)$ the anomaly score, $\lambda_\tau(q)$ the per-type threshold, and ϕ_e the Shapley value of edge e.

Table 2. Threat model: parties, assumed capabilities, and what each protection covers.

Party	Assumed capability	Protected by the design	Not protected
Aggregation server	Semi-honest: follows the protocol, inspects every message	Individual updates, feature moments, clustering statistics and score histograms (only sums are visible)	Final model, prototypes, aggregate histograms; a server that lies about the survivor set
Colluding clients	Up to $t-1$ clients share their views with the server	Inputs of honest clients, within the collusion bound; the sum itself remains visible	Sum is informative for very small K (subtraction)
Dropping clients	Up to $K-t$ clients vanish in any round	Exact sum over the surviving set	Fewer than t survivors: the round is refused
Analyst	Requests explanations for flagged nodes	Raw graph stays at the owner; explanation uses only the public model	Explanations expose the owner's own edges to the analyst
Attacker (APT)	Controls hosts, may adapt behaviour	Out of scope for the training protocol	Evasion and mimicry **[29], [30]** are not evaluated

4. Dataset

4.1 DARPA Transparent Computing Engagement 3

We use the DARPA Transparent Computing (TC) Engagement 3 (E3) corpus **[13]**, the most widely adopted public benchmark for provenance-based APT detection **[14], [31]**. Performer-instrumented hosts recorded kernel-level audit events in the Common Data Model (CDM18) format while a red team executed multi-stage attacks (browser or web-server exploit, in-memory implant, privilege escalation, reconnaissance) amid realistic benign activity. We use two hosts, **CADETS** (FreeBSD) and **THEIA** (Linux); the third host, TRACE, is excluded after an audit (§4.5). The capture is released as log files that roughly correspond to days; we call the graph built from one file a *day-graph*.

Because the raw logs occupy hundreds of gigabytes, we use the preprocessed provenance graphs distributed with the MAGIC project (MIT license) together with the entity-level ground truth of threaTrace that they carry, so that graph construction is identical to the published baselines. The parser (verified against its source) works as follows: subjects (processes) and objects (files, sockets, pipes, ...) are nodes typed by their CDM18 type, stored as integer indices because string names are not released; each audit event becomes a directed edge labelled with the event type, reversed for information-inflow events so that edges follow the information flow; at most one edge is kept per ordered node pair in a day-graph; and every edge incident to a labelled attack entity is removed from the training graphs, so that they are free of the labelled attack.

4.2 Graph Statistics

Table 3 summarises the day-graphs: each host contributes four benign training graphs and one held-out test graph containing the attack. All graphs are sparse and heavy-tailed (median degree one, three-quarters or more of the nodes are leaves, maximum degree in the hundreds of thousands), so a node's own degree and edge-type histogram are informative but its *neighbourhood* carries the discriminative signal: two leaves of identical degree can be benign or malicious depending on what they attach to.

4.3 Ground-Truth Labels

The ground truth is a list of entity identifiers, published by threaTrace from the red team's documentation that took part in the attacks; a test node is *positive* if its entity is listed (memory objects and unnamed pipes excluded). Table 4 reports the statistics. Two properties shape every result below.

Table 3. Day-graph statistics of the benign training graphs (federated clients) and the held-out test graph for THEIA and CADETS. Degree is total (in + out) degree.

Host	Split	Graph	Nodes	Edges	Mean deg.	Max deg.	Leaf frac.
THEIA	train	1	167,265	284,562	3.40	5,037	85.2%
THEIA	train	2	380,204	674,723	3.55	12,542	85.3%
THEIA	train	3	342,624	587,111	3.43	9,188	82.3%
THEIA	train	4	389,106	700,318	3.60	13,495	84.1%
THEIA	test	1	344,767	628,107	3.64	25,306	77.3%
CADETS	train	1	386,403	738,142	3.82	242,505	77.2%
CADETS	train	2	362,637	714,073	3.94	242,357	81.9%
CADETS	train	3	275,269	563,229	4.09	162,166	78.5%
CADETS	train	4	245,553	447,521	3.65	137,946	77.3%
CADETS	test	1	357,173	840,299	4.71	214,291	80.0%

Table 4. Label statistics of the test graph. Leaf fraction and mean degree compare malicious with benign nodes. Dominant type is the node-type index holding most malicious entities (count and share).

Host	Test nodes	Malicious	Rate	Malicious leaf frac.	Mean deg. mal.	Mean deg. benign	Dominant type
THEIA	344,767	25,319	7.34%	70.7%	2.37	3.74	3 (25,291; 99.9%)
CADETS	357,173	12,846	3.60%	99.7%	14.86	4.33	1 (12,806; 99.7%)

• The **positives are dominated by a single node type** (index 3 on THEIA, 1 on CADETS), so a node is compared only with benign nodes of the same type and a global threshold is inappropriate.

• On **CADETS the positives are overwhelmingly leaves** (99.7% have degree one; THEIA: 70.7%), so the node's own features carry almost no signal and the information resides in the single neighbour it attaches to.

4.4 Attack Scenarios and Threat Taxonomy

The test graphs contain the red-team campaigns of the E3 evaluation. Fig. 2 places the stages described in the public scenario descriptions on a kill chain, with the published attack windows. The technique identifiers are our own assignment of MITRE ATT&CK techniques **[1]** to the described behaviour; the data release does not carry them.

The ground truth lists *entities*, not stages, so it cannot tell us which stage a flagged node belongs to. It does allow a structural decomposition, used throughout the results to report detection per *kind of attack entity*. Let h be the malicious node with the most malicious neighbours (the *hub*) and $t^\star$ the dominant node type of Table 4. We define **A** (*fan-out leaves*): malicious degree-one nodes of type $t^\star$ attached to h; **B**: other malicious nodes of type $t^\star$; **C**: malicious nodes of the rarer types; and the hub h. Table 5 gives the counts and Table 6 the behaviour each class is consistent with. The mapping is by structure only (the released graphs carry neither entity names nor type names), so it is the interpretation most consistent with the scenario descriptions, not a measured fact.

Table 5. Anatomy of the labelled attack entities in the test graphs. Class A: degree-one malicious nodes of the dominant type attached to the hub; B: other malicious nodes of the dominant type; C: malicious nodes of other types. Max. deg. is the largest degree among the malicious entities.

Host	Malicious	Hub degree	A fan-out	B other	C rare	Max. deg.
THEIA	25,319	25,306	17,897	7,394	28	25,306
CADETS	12,846	12,839	12,788	18	40	47,418

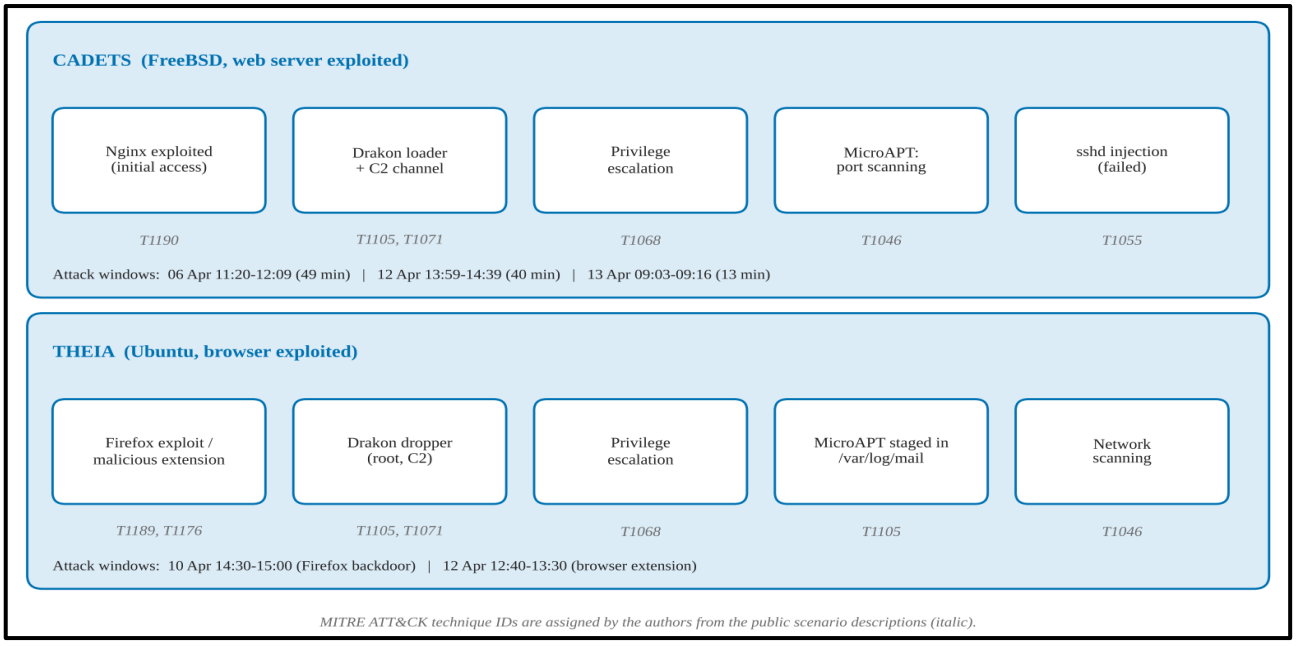


Figure 2. Attack chains of the CADETS (top) and THEIA (bottom) scenarios with the author-assigned MITRE ATT&CK techniques below each stage and the published attack windows.

Table 6. Attack-entity classes, their structural signature, and the behaviour they are consistent with (author interpretation).

Class	Structural signature	Consistent with	ATT&CK
A: fan-out leaves	Degree-one nodes of the dominant type, one edge type, all attached to the hub	Network scanning: one implant process touching thousands of endpoints	T1046
B: other dominant type	Dominant-type nodes in the hub neighbourhood that are not its leaves	Scan targets and staged artefacts with follow-up activity	T1046, T1105
C: rare types	Processes, files and other entity types of low frequency	Exploited application, loader and dropped payload, command-and-control channel	T1190, T1189, T1105, T1071, T1068
Hub	Highest-degree malicious process	Implant process driving the scan and the C2 channel	T1055, T1071

4.5 Train/Test Overlap Audit and the Exclusion of TRACE

Before using a host we audit whether its train and test graphs are independent, comparing degree and edge-type histograms of the training and test graph (normalised ℓ_1 distance) and the node/edge count difference (Table 7). On THEIA and CADETS the test day differs from every training day and the distances are large. On TRACE the first four test graphs are built from *the same log files* as the training graphs, differing only by the removal of attack-incident edges (node counts differ by five, degree-histogram distance below 0.003), so evaluation would be effectively transductive. We therefore exclude TRACE from all headline detection results and report this audit instead.

Table 7. Train/test independence audit. ℓ_1 distances between normalised degree and edge-type histograms of the training graph and the test graph at the same position; node and edge differences are test minus train. TRACE test graphs reuse the training logs.

Host	Pair	Degree-hist. ℓ_1	Edge-type-hist. ℓ_1	Node diff.	Edge diff.
THEIA	1	0.1731	0.2066	+177,502	+343,545
CADETS	1	0.1727	0.1632	-29,230	+102,157
TRACE	1	0.0028	0.0153	+5	+9,205
TRACE	2	0.0023	0.0128	+5	+6,076
TRACE	3	0.0011	0.0075	+5	+3,157
TRACE	4	0.0007	0.0065	+756	+2,629

4.6 Federated Partition

The four benign day-graphs of a host are the four clients ($K = 4$), a *temporally partitioned, non-IID* federation (167,265–389,106 nodes on THEIA, 245,553–386,403 on CADETS, different degree extremes and edge-type mixtures). It does not model heterogeneity between *organisations* (§8.5). The test graph is never seen by any client, by the server, or during threshold calibration.

5. Methodology

5.1 System Overview

Fig. 3 shows the system. Each client converts its audit stream into a provenance graph (Fig. 4) and trains a shared FedGAT model locally on a masked-feature objective; in every round it sends its update in *masked* form, so the server obtains only the average. The same secure-sum primitive is then reused to agree on feature statistics, fit benign prototypes per node type, and calibrate per-type alarm thresholds. At deployment a node's embedding is compared with the prototypes of its type to produce the anomaly score, and a flagged node is passed to the Shapley explainer.

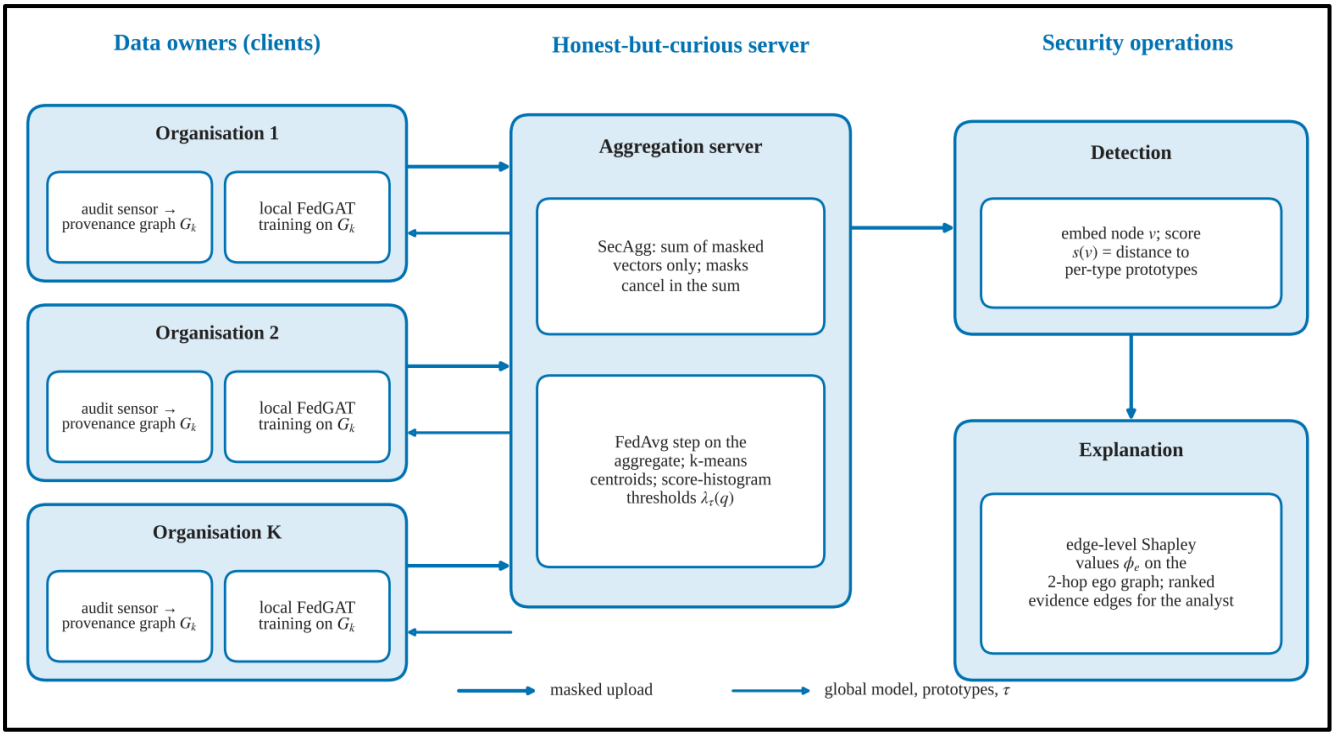


Figure 3. System architecture. Clients keep their provenance graphs; the server sees only securely aggregated sums (model updates, clustering statistics, score histograms). Detection and edge-level explanation run on the shared model and prototypes.

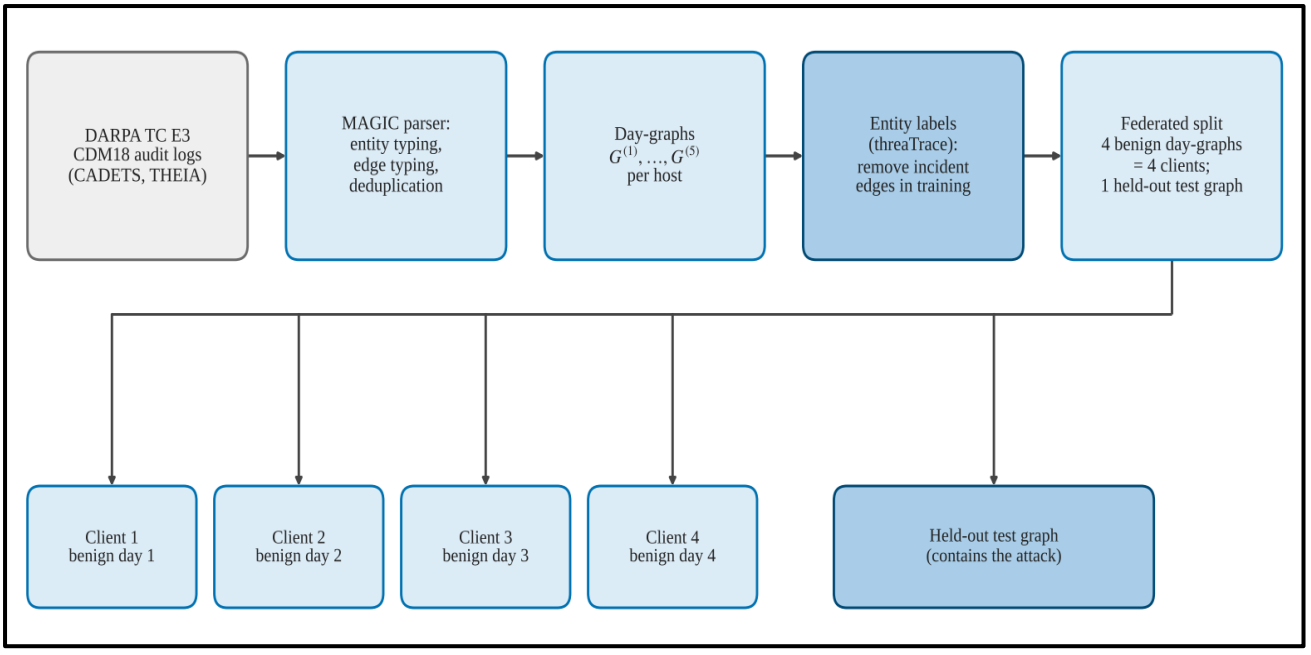


Figure 4. Data preparation from the DARPA TC E3 logs to federated clients, following the MAGIC release.

5.2 Node Features and Federated Standardization

The released graphs carry only integer node and edge types, so node features are derived structurally. For node v let $c_{v,r}^{\text{out}} = |\{(v,w,r) \in E\}|$ and $c_{v,r}^{\text{in}} = |\{(u,v,r) \in E\}|$ be its per-type out- and in-edge counts. The feature vector is

$$x_v = \left[\text{onehot}\big(\tau(v)\big) \,\|\, \log\big(1 + c_{v,\cdot}^{\text{out}}\big) \,\|\, \log\big(1 + c_{v,\cdot}^{\text{in}}\big) \,\|\, \log(1+\| c_v^{\text{out}} \|_1) \,\|\, \log\big(1+\| c_v^{\text{in}} \|_1\big)\right] \in \mathbb{R}^d, \tag{2}$$

With $d = K_n + 2K_e + 2$ (41 for THEIA, 62 for CADETS). Features are standardised with *global* statistics that no client can compute alone. Each client k with n_k nodes submits the vector $(n_k, \sum_v x_v, \sum_v x_v \odot x_v)$ to the secure sum, and the server derives

$$\mu = \frac{\sum_k \sum_v x_v}{\sum_k n_k}, \qquad \sigma^2 = \frac{\sum_k \sum_v x_v \odot x_v}{\sum_k n_k} - \mu \odot \mu, \qquad \tilde{x}_v = \frac{x_v - \mu}{\sigma + \varepsilon}, \tag{3}$$

With $\varepsilon = 10^{-3}$. Only the pooled moments (μ, σ) are revealed. The same (μ, σ) are applied to the test graph. Fig. 5 illustrates the layout of the feature vector.

5.3 Edge-Typed GATv2 Encoder

The encoder is a two-layer graph attention network in the GATv2 form **[37]**, extended with relation-type embedding's in the spirit of R-GCN **[35]** so that a "write" edge and a "connect" edge are not interchangeable. Let $\epsilon_r \in \mathbb{R}^{16}$ be the learned embedding of relation $r \in \{1, \ldots, 2K_e\}$ (forward types and their reverses), and let $h_v^{(0)} = \text{ELU}(W_{\text{in}} \tilde{x}_v)$ with $W_{\text{in}} \in \mathbb{R}^{h \times d}$ and $h = 64$. For layer $\ell \in \{1,2\}$ and head $a \in \{1, \ldots, H\}$ ($H = 4$, $C = h/H = 16$) the message along $(u,r) \in \mathcal{N}(v)$ and the unnormalised attention logit are

$$m_{u \to v}^{(\ell,a)} = W_l^{(\ell,a)} h_u^{(\ell-1)} + W_e^{(\ell,a)} \epsilon_r, \tag{4}$$

$$\eta_{uv}^{(\ell,a)} = \mathbf{a}^{(\ell,a)\top} \text{LeakyReLU}_{0.2}\big(m_{u \to v}^{(\ell,a)} + W_r^{(\ell,a)} h_v^{(\ell-1)}\big), \tag{5}$$

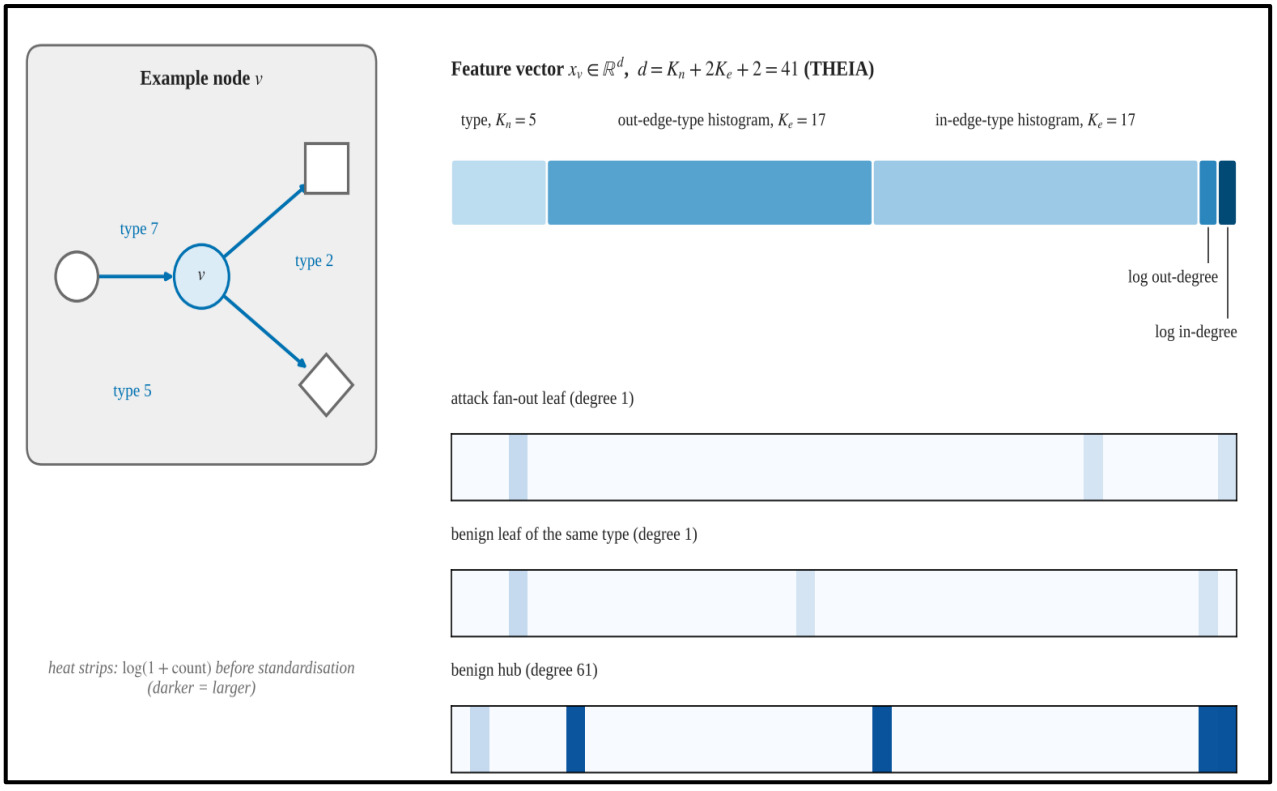


Figure 5. Structural node features. Left: an example node with typed incident edges. Right: layout of the feature vector of (2) for THEIA ($d = 41$) and heat strips of the log-count features of an attack fan-out leaf, a benign leaf of the same type and a benign hub; both leaves have a single active relation, so their own features carry little information.

Where $\mathbf{a}^{(\ell,a)} \in \mathbb{R}^C$ is the attention vector. Applying the non-linearity *before* the dot product with $\mathbf{a}$ is what makes the attention dynamic, i.e. the ranking of neighbours may depend on the query node **[37]**. The attention coefficients are a softmax over the sampled in-neighbourhood, and the layer output concatenates the heads, adds a root (self) transformation, and applies layer normalisation and ELU:

$$\alpha_{uv}^{(\ell,a)} = \frac{\exp \eta_{uv}^{(\ell,a)}}{\sum_{(w,r)\in\mathcal{N}(v)} \exp \eta_{wv}^{(\ell,a)}}, \tag{6}$$

$$h_v^{(\ell)} = \mathrm{ELU}\left(\mathrm{LN}\left(\Big\|_{a=1}^{H} \sum_{(u,r)\in\mathcal{N}(v)} \alpha_{uv}^{(\ell,a)}\, m_{u\to v}^{(\ell,a)} + W_{\mathrm{root}}^{(\ell)} h_v^{(\ell-1)} + b^{(\ell)}\right)\right). \tag{7}$$

The final embedding is $z_v = h_v^{(2)} \in \mathbb{R}^{64}$. The decoder is a two-layer MLP, $\hat{x}_v = g_\varphi(z_v) = W_2\,\mathrm{ELU}(W_1 z_v + b_1) + b_2$. The whole model has 37,778 trainable parameters on THEIA and 41,148 on CADETS (Fig. 6), small enough that a full update is about 150 KB in single precision.

5.4 Masked-Feature Self-Supervision and Sampling

No attack labels are used. Following masked autoencoding for graphs **[38]**, a batch $\mathcal{B}$ of seed nodes has its own standardised features replaced by a learned mask token $\mathbf{m} \in \mathbb{R}^d$ while their neighbours stay visible, and the model reconstructs the hidden features from the neighbourhood:

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|\, d} \sum_{v\in\mathcal{B}} \| g_\varphi\big(z_v(\tilde{x}_v \leftarrow \mathbf{m})\big) - \tilde{x}_v \|_2^2. \tag{8}$$

Minimizing (8) on benign graphs teaches the encoder what a node *should* look like given its context; a node with atypical context is mapped to a region that benign nodes do not occupy, which is the signal the scorer exploits. At test time no masking is applied. Because degrees reach 10^5, we sample at most f_ℓ incoming relations per layer with fan-outs $(f_1, f_2) =$

(10,5) (uniformly with replacement for larger degrees), giving at most $1 + f_1 + f_1 f_2 = 61$ nodes per target; scoring uses a fixed sampling seed, so the score is a deterministic function of the model and the graph.

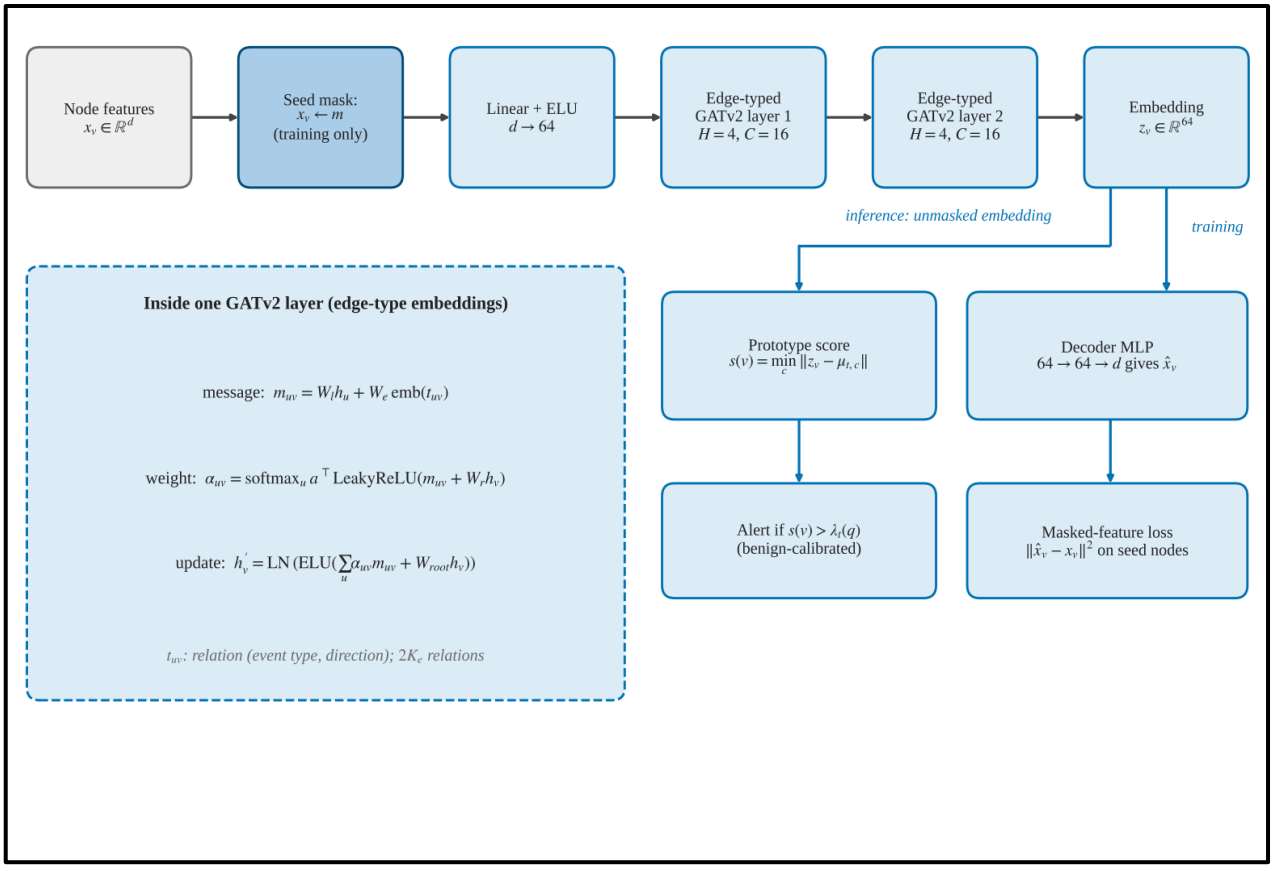


Figure 6. FedGAT encoder-decoder. The seed node's own features are replaced by a learned mask token during training; scoring uses the unmasked embedding.

5.5 Federated Optimisation

Round t of FedAvg **[15]** starts from global parameters θ^t. Every client k runs $\tau = 25$ Adam steps (up to $B = 2048$ seed nodes, learning rate 2×10^{-3}, optimiser state reset each round), producing $\Delta_k^t = \theta_k^{t,\tau} - \theta^t$, and the server forms

$$\theta^{t+1} = \theta^t + \frac{1}{|U_3|} \sum_{k \in U_3} \Delta_k^t, \tag{9}$$

Where U_3 is the set of clients whose update arrived. Clients are weighted equally rather than by n_k (sizes differ by at most a factor of two, and equal weights avoid revealing n_k-dependent information outside the secure sum). The FedProx variant **[39]** adds $\frac{\mu}{2} \| \theta - \theta^t \|_2^2$ with $\mu = 0.01$. We run $T = 30$ rounds ($T\tau K = 3000$ gradient steps in aggregate); the centralised reference uses the same number of steps with a single persistent Adam state.

Algorithm 1 *Federated training of FedGAT under secure aggregation (one run)*

```
Input: clients k=1..K with graphs G_k; rounds T; local steps tau; threshold t = floor(K/2)+1
(mu, sigma) <- SecSum({n_k, sum x, sum x*x})              # global standardisation
server initialises theta^0
for round r = 1..T do
    server broadcasts theta^r
    for each client k in parallel do
        theta_k <- theta^r;  run tau Adam steps on masked-feature loss (Eq. loss) over G_k
        Delta_k <- theta_k - theta^r;  send Mask(Delta_k) to the server      # Algorithm 2
    end for
    S <- SecSum({Delta_k : k in U3});  theta^{r+1} <- theta^r + S/|U3|
end for
return theta^T
```

5.6 Secure Aggregation Protocol

All cross-client quantities are aggregated with a dropout-tolerant protocol following Bonawitz et al. **[19]**, restricted to the semi-honest server model (Fig. 7, Table 8). A client vector $x_u \in \mathbb{R}^m$ is converted to fixed point in the ring $\mathbb{Z}_{2^{64}}$,

$$q(x) = \left(\text{round}\left(2^s \cdot \text{clip}(x, -c, c)\right)\right) \bmod 2^{64}, \tag{10}$$

With $s = 24$, $c = 64$ for model updates ($s = 16$, $c = 10^{12}$ for moment vectors; histogram counts are integers, $s = 0$).

Setup and key sharing**.** Client u draws two X25519 key pairs (one for share transport, one for mask agreement) and a random self-mask seed b_u; for every pair a *pairwise seed* is derived symmetrically,

$$s_{uv} = \text{HKDF}\left(\text{X25519}(s_u^{\text{SK}}, s_v^{\text{PK}})\right) = s_{vu}. \tag{11}$$

Client u splits s_u^{SK} and b_u with Shamir's t-out-of-K scheme **[44]** over $\mathbb{F}_p$, $p = 2^{521} - 1$ (reconstruction is Lagrange interpolation at zero), and sends each share pair encrypted with AES-256-GCM through the server. Masked input and unmasking**.** Client u uploads

$$y_u = q(x_u) + \text{PRG}(b_u) + \sum_{v \in U_2, v \neq u} \sigma\,(u, v)\,\text{PRG}(s_{uv}) \pmod{2^{64}}, \tag{12}$$

Where $\sigma(u, v) = +1$ if $u > v$ and -1 otherwise, PRG expands a 256-bit seed into m ring elements with ChaCha20, and U_2 is the set of clients that completed key sharing; the server fixes the survivor set $U_3 \subseteq U_2$. Survivors then return shares of b_u for $u \in U_3$ and of s_u^{SK} for dropped u (never both for the same peer), and the server computes

$$\sum_{u \in U_3} q\,(x_u) = \sum_{u \in U_3} y_u - \sum_{u \in U_3} \text{PRG}\,(b_u) - \sum_{d \in U_2 \setminus U_3} \sum_{v \in U_3} \sigma\,(v, d)\,\text{PRG}(s_{vd}), \tag{13}$$

Returning 2^{-s} signed $\left(\sum_{u \in U_3} q\,(x_u)\right)$.

Table 8. Rounds of the secure aggregation protocol with the payload each client sends (K clients, vector length m, 64-bit ring).

Round	Sender	Content	Client bytes	Purpose
R0	Client	Two X25519 public keys	64	Establish pairwise secrets
R1	Client	K AES-GCM encrypted share pairs	$160K$	Dropout recovery
R2	Client	Masked vector $y_u \in \mathbb{Z}_{2^{64}}^m$	$8m$	Contribute the update
R3	Server	Survivor list U_3	0 (downlink)	Fix the summed set
R4	Client	Shares of b_u (survivors) or s_u^{SK} (dropouts)	$66K$	Remove the masks

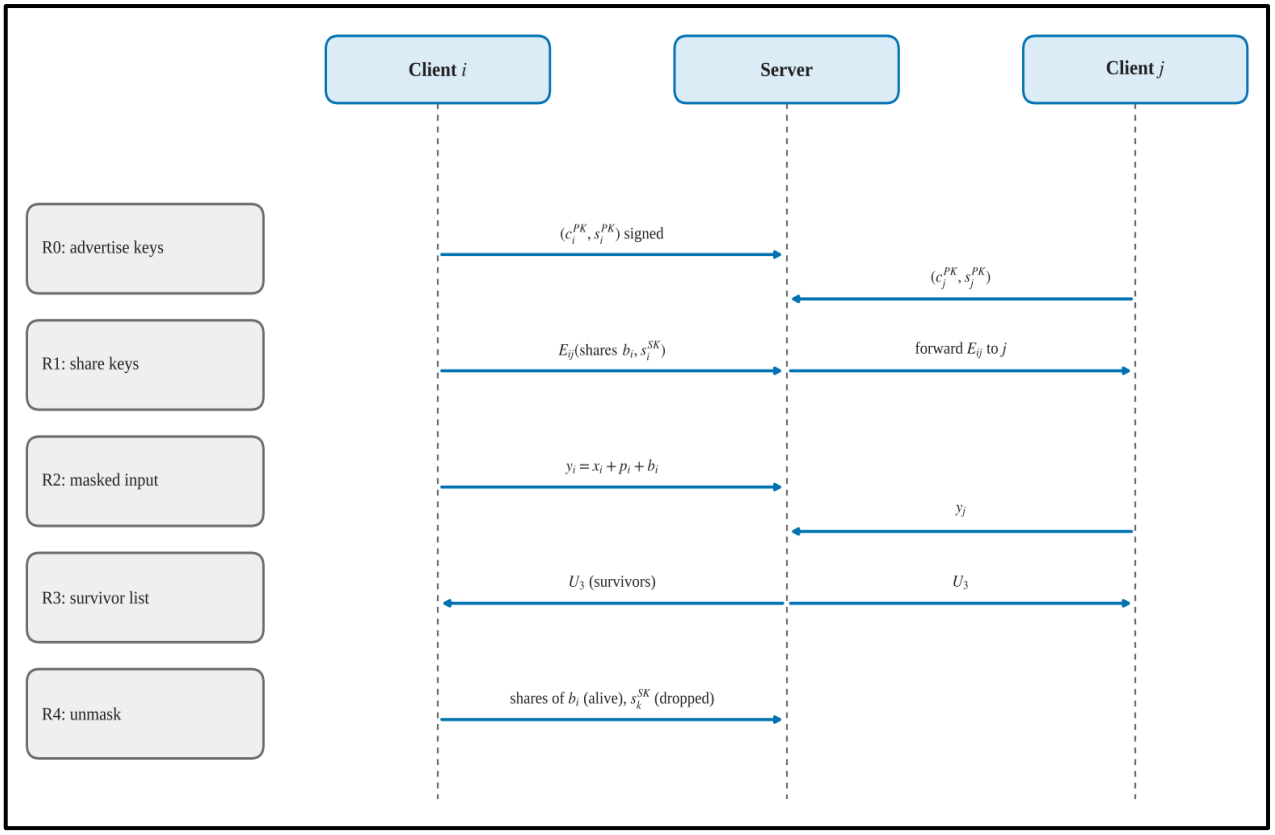


Figure 7. Message flow of the secure aggregation protocol. R0: key advertisement; R1: encrypted Shamir shares; R2: masked input; R3: survivor list; R4: share reconstruction. The server obtains only the sum over the survivors.

Correctness. For $u \neq v$ both in U_3 the pairwise masks cancel by (11); self masks are removed explicitly; for a client d that dropped before round 2, the server reconstructs s_d^{SK} from t shares and subtracts the unmatched terms $\sigma(v,d)\mathrm{PRG}(s_{vd})$ of the survivors. The output therefore equals $\sum_{u\in U_3} q\,(x_u)$ in $\mathbb{Z}_{2^{64}}$.

Proposition 1 (exactness up to quantisation). If $|x_{u,i}| \leq c$ for all clients and coordinates and $|U_3|\,c\,2^s < 2^{63}$ (no ring overflow), then the secure sum returned to the server differs from the true sum by at most $|U_3|\,2^{-s-1}$ per coordinate.

Proof. The ring sum equals $\sum_u q\,(x_u)$ exactly. Each $q(x_{u,i})$ differs from $2^s x_{u,i}$ by at most $1/2$ from rounding, so after division by 2^s the error per client is at most 2^{-s-1}, and the errors add at most linearly over $|U_3|$ clients. The overflow condition guarantees that the signed interpretation of the ring element is the integer sum. ▫

With $K = 4$, $s = 24$, $c = 64$ the bound is $4 \cdot 2^{-25} \approx 1.2 \times 10^{-7}$ and the overflow margin is $4 \cdot 64 \cdot 2^{24} = 2^{32} \ll 2^{63}$. The measured maximum deviation is reported in §7.7.

5.7 Federated Prototypes and Threshold Calibration

The detector must turn embeddings into scores and scores into alarms without any client revealing its embeddings, so we use *per-node-type k-means prototypes* and *per-type score histograms*, both of which are sums (Fig. 8). A kNN scorer as in MAGIC is not federation-compatible; on the same embeddings (centralised) the prototype scorer matched it (THEIA 0.997 versus 0.997; CADETS 0.955 versus 0.958 AUROC).

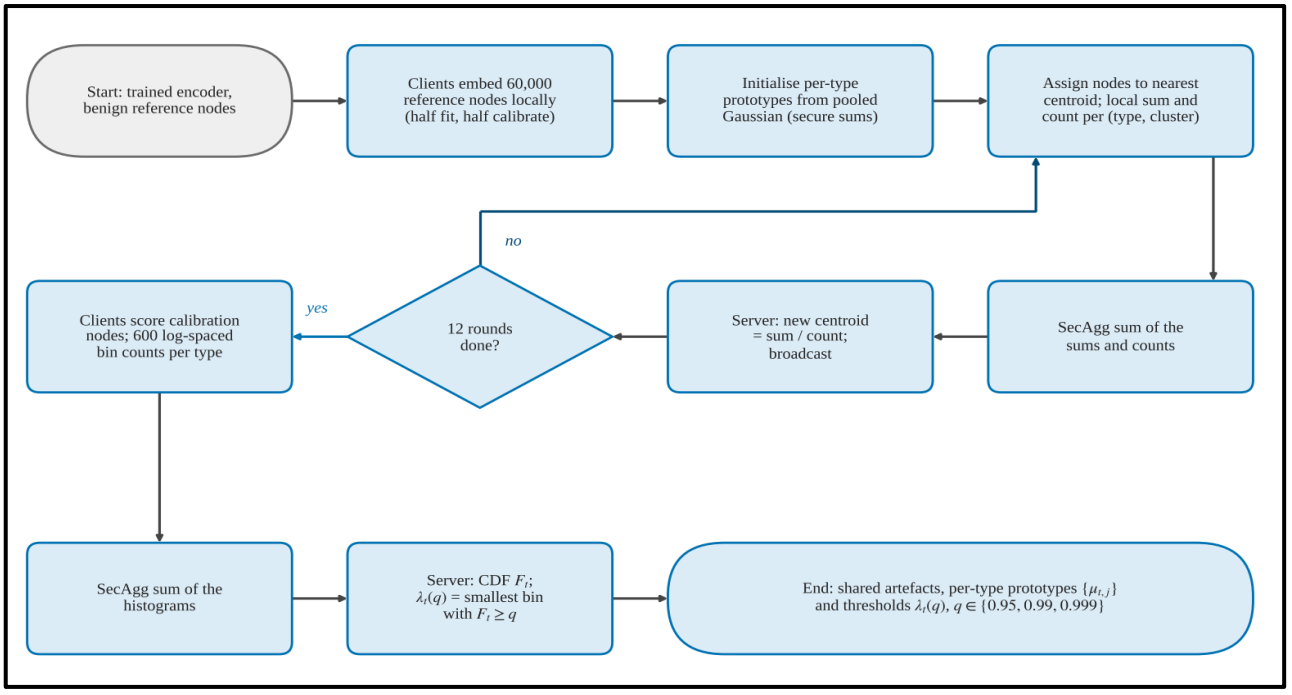


Figure 8. Federated prototype fitting and threshold calibration as a flowchart. Every client-to-server step is a securely aggregated sum.

Fit/calibration split. Each client embeds up to $n_{\text{ref}} = 60{,}000$ benign nodes with the final global model and splits them equally into a *fit* half and a *calibration* half. **Initialisation.** Per node type τ the clients securely sum count, mean and variance, giving N_τ, m_τ, ς_τ; $K_\tau = \max\big(1,\min(K, \lfloor N_\tau/20 \rfloor)\big)$ with $K = 256$ and

$$c_{\tau,j}^{(0)} = m_\tau + 0.7\,\varsigma_\tau \odot \xi_j, \qquad \xi_j \sim \mathcal{N}(0, I). \tag{14}$$

Federated Lloyd iterations. For 12 iterations every client assigns its fit embeddings to the nearest prototype and submits per-prototype sums $S_{\tau,j}^k$ and counts $n_{\tau,j}^k$; the server updates

$$c_{\tau,j} \leftarrow \frac{\sum_k S_{\tau,j}^k}{\sum_k n_{\tau,j}^k} \quad \text{if } \sum_k n_{\tau,j}^k > 0.5, \text{otherwise unchanged.} \tag{15}$$

Score and calibration. The anomaly score is the distance to the nearest prototype of the node's own type, computed from the unmasked embedding,

$$s(v) = \min_{1 \le j \le K_{\tau(v)}} \| z_v - c_{\tau(v),j} \|_2. \tag{16}$$

Each client histograms its calibration scores over 600 log-spaced bins $e_b = 10^{-4+6.5\,b/600}$ per type and submits them through the secure sum; the pooled empirical CDF F_τ gives the threshold at benign quantile q:

$$\lambda_\tau(q) = \min\{e_b : F_\tau(e_b) \ge q\}, \qquad \hat{y}_v = \mathbf{1}\big[s(v) > \lambda_{\tau(v)}(q)\big]. \tag{17}$$

The operating points $q \in \{0.95, 0.99, 0.999\}$ correspond to nominal benign alarm rates of 5%, 1% and 0.1% per type. They use no attack information, so the reported precision, recall and false-positive rate are those a deployed system would obtain.

5.8 Edge-Level Shapley Attribution

For a flagged node v we want to know *which provenance edges* produce its score. We cast this as a cooperative game and compute the Shapley value of each edge. Fig. 9 shows how the explainer attaches to the shared model: it needs only public artefacts and queries the model repeatedly under different coalitions.

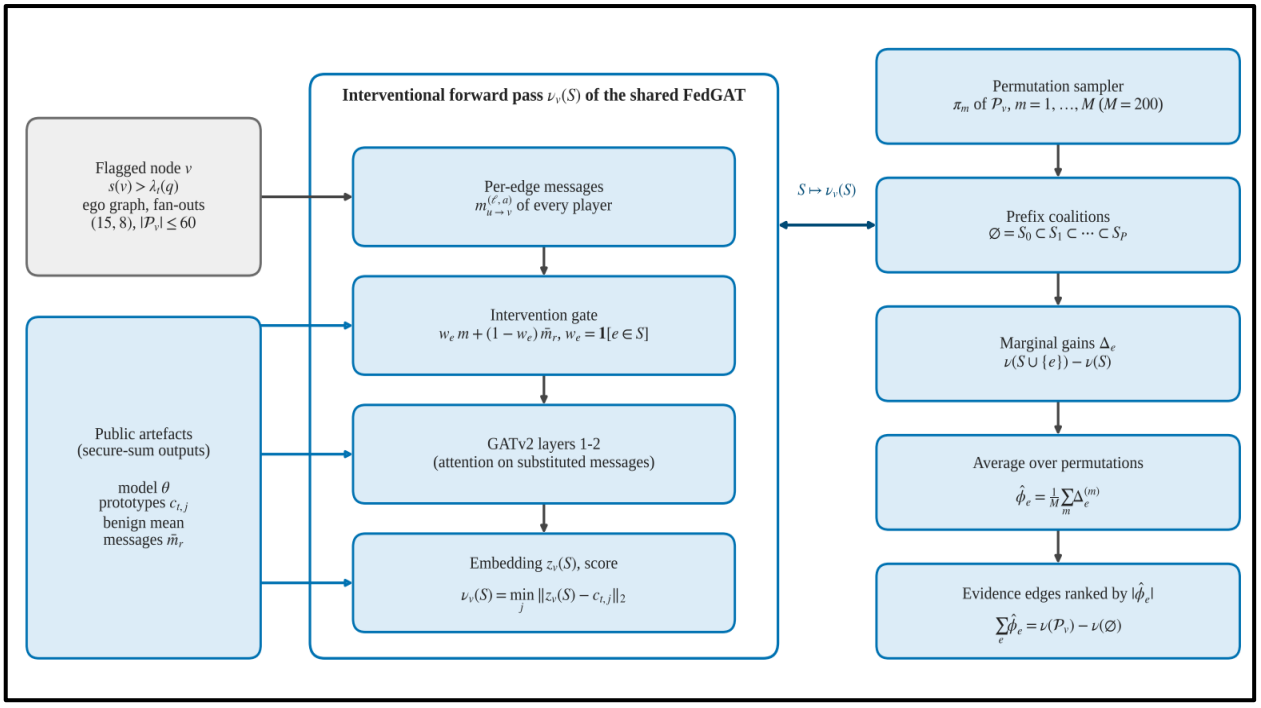


Figure 9. Architecture of the SHAP-based edge explainer. The ego graph of a flagged node supplies the players; each coalition query runs the shared FedGAT with the messages of absent edges replaced by benign mean messages; the permutation sampler turns the resulting scores into marginal gains and averages them into edge attributions.

Players. The players are the undirected provenance edges of the 2-hop ego graph of v (fan-outs (15,8), capped at $|\mathcal{P}_v| \le 60$); an edge stands for both message directions, so removing a player removes the information flow in both.

Interventional value function. Simply deleting an edge changes the node's degree context and moves the embedding far outside the benign distribution (an isolated node has $\nu(\emptyset)$ far above $\nu(\mathcal{P}_v)$ in our model, so every explanation would trivially read "all edges reduce the score"). We therefore replace the *message* of an absent edge by the benign mean message of its relation type (Fig. 10):

$$\tilde{m}^{(\ell,a)}_{u\to v}(w_e) = w_e\, m^{(\ell,a)}_{u\to v} + (1 - w_e)\, \bar{m}^{(\ell,a)}_r, \qquad w_e = \mathbf{1}[e \in S], \tag{18}$$

Where $\bar{m}^{(\ell,a)}_r$ is the mean message of relation r in layer ℓ over benign training nodes (in a deployment, one more securely aggregated mean). Attention logits are computed from the substituted messages. Writing $z_v(S)$ for the resulting embedding,

$$\nu_v(S) = \min_j \| z_v(S) - c_{\tau(v),j} \|_2. \tag{19}$$

So $\nu_v(\emptyset)$ is the score of a node whose entire neighbourhood is benign-typical and $\nu_v(\mathcal{P}_v)$ the actual score. This is an *interventional* value function.

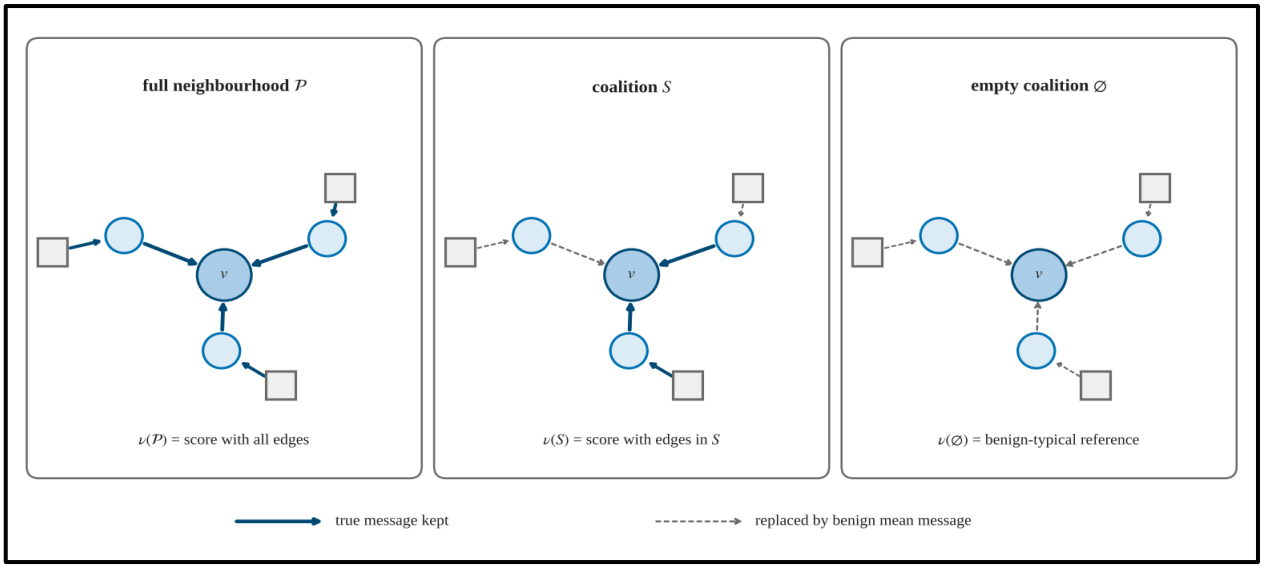


Figure 10. Interventional value function. A coalition S keeps the true message of its edges (solid, red) and replaces all other edge messages by the benign per-relation mean message (dashed). $\nu(\mathcal{P})$ is the detector's actual score and $\nu(\varnothing)$ a benign-typical reference.

Shapley value and estimator. The Shapley value of edge e, and its permutation-sampling estimate, are

$$\phi_e = \sum_{S \subseteq \mathcal{P}_v \setminus \{e\}} \frac{|S|!\,(|\mathcal{P}_v| - |S| - 1)!}{|\mathcal{P}_v|!} [\nu_v(S \cup \{e\}) - \nu_v(S)] = \mathbb{E}_\pi[\nu_v(\mathrm{Pre}_\pi(e) \cup \{e\}) - \nu_v(\mathrm{Pre}_\pi(e))], \tag{20}$$

$$\hat{\phi}_e = \frac{1}{M} \sum_{m=1}^{M} \left[\nu_v(\mathrm{Pre}_{\pi_m}(e) \cup \{e\}) - \nu_v(\mathrm{Pre}_{\pi_m}(e))\right], \qquad M = 200. \tag{21}$$

Where π is a uniformly random permutation and $\mathrm{Pre}_\pi(e)$ the players before e.

Proposition 2 (exact efficiency for any number of permutations). For every $M \geq 1$ and every realisation of the permutations, $\sum_{e \in \mathcal{P}_v} \hat{\phi}_e = \nu_v(\mathcal{P}_v) - \nu_v(\emptyset)$.

Proof. Within one permutation the marginal contributions telescope: $\sum_e \left[\nu(\mathrm{Pre}_\pi(e) \cup \{e\}) - \nu\big(\mathrm{Pre}_\pi(e)\big)\right] = \nu(\mathcal{P}_v) - \nu(\emptyset)$. Averaging over the M permutations preserves the identity. ▫

Proposition 3 (concentration). If every marginal contribution lies in $[-R, R]$, then $\Pr\big(|\hat{\phi}_e - \phi_e| \geq \varepsilon\big) \leq 2\exp\big(-M\varepsilon^2/(2R^2)\big)$.

Proof. The M marginal contributions are i.i.d. with mean ϕ_e and range $2R$; apply Hoeffding's inequality. ▫

The Shapley value is the *unique* attribution that is efficient, symmetric, additive and gives zero to dummy players. Since ν_v is a score, $\phi_e > 0$ means edge e *increases* the anomaly relative to a benign-typical replacement; for ranking evidence we use $|\phi_e|$.

5.9 Attribution Baselines and Fidelity Metrics

We compare the explainer with five baselines evaluated on the *same* players and value function: Random (uniform ranking); Attention (coefficient averaged over heads and layers, summed over both directions); Saliency (absolute gradient of the score with respect to a per-edge weight w_e at 1); a GNNExplainer-style mask [22] (soft mask optimised for 80 Adam steps with ℓ_1 penalty 0.02, a lightweight variant); and Occlusion ($|\nu(\mathcal{P}_v) - \nu(\mathcal{P}_v \setminus \{e\})|$). Only Shapley is complete (efficient) and signed; Occlusion and Saliency are signed but not complete.

Deletion and insertion fidelity. With T_k the k highest-ranked edges, deletion measures how much of the node's anomalous excess disappears when they are replaced by benign-typical messages, and insertion how much is recovered by keeping only them:

$$D(k) = \frac{\nu(\mathcal{P}_v) - \nu(\mathcal{P}_v \setminus T_k)}{\nu(\mathcal{P}_v) - \nu(\emptyset)}, \qquad I(k) = \frac{\nu(T_k) - \nu(\emptyset)}{\nu(\mathcal{P}_v) - \nu(\emptyset)}. \tag{22}$$

Both are reported at $k \in \{1,2,3,4,6,8\}$ and as the mean over these k (an analyst reads a fixed number of edges). Nodes are admitted only if the excess $\nu(\mathcal{P}_v) - \nu(\emptyset)$ exceeds 10^{-3} and the ego graph has at least eight players, three incident to v.

Ground-truth alignment and stability. Each player carries four binary labels from the published entity labels: *incident*, *any-malicious*, *both-malicious*, and (incident edges only) *malicious-neighbour*; we report the AUROC of $|\phi_e|$ against each, averaged over nodes. These labels are entity-level, not event-level, so alignment complements fidelity and is never the sole criterion. Monte-Carlo convergence is measured for $M \in \{5,10,25,50,100,200\}$ against an $M = 800$ reference by Spearman rank correlation, top-3 overlap and efficiency gap.

5.10 Complexity and Security Analysis

Complexity. A local training step costs $O(Bf_1f_2h^2)$, independent of graph size. A secure sum costs each client $O(K)$ key agreements and $O(Km)$ PRG output and uploads $64 + 160K + 8m + 66K$ bytes, i.e. about twice a float32 vector (64-bit ring) plus a K-dependent term that vanishes for large m. One attribution costs $M|\mathcal{P}_v|$ forward passes on at most 60 edges. Table 9 summarises the costs.

Table 9. Asymptotic cost per component (K clients, vector length m, hidden size h, batch B, fan-outs f_1, f_2, M permutations, P players).

Component	Client computation	Server computation	Communication per client
Local training step	$O(Bf_1f_2h^2)$	none	none
Secure sum (one call)	$O(Km)$ PRG output	$O(Km)$ additions	$8m + 226K + 64$ bytes
Prototype round	local assignment, $O(n_k c\, h)$	$O(Kc\, h)$	secure sum of $c(h+1)$ values per type
Calibration	local binning	$O(K \cdot \text{bins})$	one secure sum of 600 counts per type
Explanation	$O(MP^2h)$	none	none

What is protected? Under the semi-honest model with up to $t-1$ colluding parties, the server's view of y_u is $q(x_u)$ plus the pseudorandom mask $\text{PRG}(b_u)$, which stays independent of the server's knowledge unless b_u is reconstructed; b_u is reconstructed only for $u \in U_3$ after the input has been added into the sum, and never together with s_u^{SK}. The server therefore learns the sum over U_3 and nothing more about an individual x_u; §7.7 tests this empirically.

What is not protected? (i) A malicious server that lies about U_3 breaks the guarantee; preventing this needs the signature-based consistency round of the full protocol, which we do not implement. (ii) A server can shape the downlink models so that the aggregate isolates one client **[27]**. (iii) With $K = 4$, a coalition controlling the other three clients recovers the fourth update by subtraction, so the guarantee is meaningful only within the collusion bound $t-1$ and strengthens with K. (iv) The sums are not differentially private **[28]** (§8.5).

6. Experimental Setup

6.1 Implementation and Environment

Every component is implemented in Python with PyTorch (typed graph store, neighbour sampler, edge-typed GATv2 layers, federated loops, federated k-means and calibration, Shapley explainer). The secure aggregation layer uses the cryptography package for X25519, HKDF-SHA256, AES-256-GCM and ChaCha20 and our own Shamir implementation over $\mathbb{F}_{2^{521}-1}$. Experiments run on a CPU-only container (two Xeon cores at 2.8 GHz, 7 GB RAM; Python 3.13, PyTorch 2.15); two single-threaded training runs execute concurrently, so training times are upper bounds, whereas the secure-aggregation benchmarks of §7.7 ran on an idle machine. Clients and server are simulated in one process but exchange only the byte strings and vectors the protocol prescribes.

6.2 Hyperparameters

Table 10 lists all settings, fixed before the experiments and not tuned on test labels with one disclosed exception: the prototype scorer and $K = 256$ were chosen in exploratory runs that computed AUROC on the test graphs (§8.5). All hyperparameters are shared by the federated and centralised configurations, so comparisons between configurations are unaffected.

Table 10. Hyperparameters.

Component	Setting	Component	Setting
Encoder	2 GATv2-Edge layers, $h = 64$, $H = 4$, $C = 16$	Edge-type embedding	16 dimensions, $2K_e$ relations
Neighbour fan-out	$(10,5)$, with replacement above degree	Decoder	MLP $64 \to 64 \to d$, ELU
Optimiser	Adam, lr 2×10^{-3}	Batch	up to 2,048 distinct seed nodes
Clients	$K = 4$ (benign day-graphs)	Rounds / local steps	$T = 30$ / $\tau = 25$
FedProx	$\mu = 0.01$	Shamir threshold	$t = \lfloor K/2 \rfloor + 1 = 3$
Update quantisation	$s = 24$, $c = 64$	Statistic quantisation	$s = 16$, $c = 10^{12}$
Prototypes	$K = 256$ per type, 12 Lloyd iterations	Reference nodes	60,000 per client (half fit, half calibrate)
Histogram	600 log-spaced bins, 10^{-4} to $10^{2.5}$	Operating points	$q \in \{0.95, 0.99, 0.999\}$
Shapley	$M = 200$ permutations, ego fan-outs $(15,8)$, at most 60 players	Seeds	0, 1, 2 (core configurations)

6.3 Compared Configurations

All configurations share the architecture, the number of gradient steps and the scoring pipeline; they differ in *how* the model and statistics are obtained.

- **Centralized.** The four graphs are pooled and one model is trained for the same $T\tau K$ steps with a persistent Adam state; the statistics, prototypes and thresholds are computed in the clear. This is a privacy-free reference, not a deployable option.
- **FedAvg (plain).** Federated averaging with all aggregates computed in the clear on the server. It isolates the effect of federation from the effect of secure aggregation.
- **FedAvg + SecAgg (proposed).** As above, with *every* cross-client sum (standardisation moments, model updates, k-means statistics, histograms) computed with the real protocol of §5.6.
- **FedProx + SecAgg.** The proximal variant ($\mu = 0.01$).
- **Local only.** Each client trains alone for the same number of local steps and builds its own prototypes and thresholds; we report the mean and standard deviation over the four clients. This quantifies what federation adds.
- **Raw features (no GNN).** The same prototype scorer applied directly to the standardised input features, without any learned embedding. It shows how much of the signal is available before any graph learning.
- **Ablations (FedAvg + SecAgg):** GraphSAGE-style uniform aggregation (attention logits fixed to zero, i.e. mean over messages); no edge-type information (all relations share one embedding); no masking (the seed's features remain visible); no graph (an MLP that sees only the node's own features).

6.4 Metrics

The primary ranking metrics are the area under the ROC curve and the average precision (area under the precision-recall curve). With v_{p} and v_{n} drawn uniformly from the malicious and benign test nodes, ties counting one half,

$$\mathrm{AUROC} = \Pr\left(s(v_{\mathrm{p}}) > s(v_{\mathrm{n}})\right) + \frac{1}{2}\Pr\left(s(v_{\mathrm{p}}) = s(v_{\mathrm{n}})\right). \tag{23}$$

Because the positives are 2–7% of the nodes, AUROC alone is optimistic, so we also report the true-positive rate at a 1% and 5% false-positive rate, the precision among the top 100 and top 1,000 ranked nodes, and the *oracle* F1, the best F1 over all possible thresholds (an upper bound on what any threshold policy can achieve with the given scores). The *deployable* operating points use the benign-calibrated thresholds $\lambda_\tau(q)$ of (17) and report

$$P = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \qquad R = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \qquad F_1 = \frac{2PR}{P+R}, \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}. \tag{24}$$

Results of the core configurations are reported as the mean $\pm$ the sample standard deviation over seeds; the seed controls the model initialisation, batch sampling, neighbour sampling and the k-means initialisation. Explanation metrics are averaged over nodes with a bootstrap 95% confidence interval for paired differences between methods.

6.5 Protocols

Detection. Models are trained on the four benign day-graphs and evaluated on the held-out test graph of the same host. The score of every test node is computed (no subsampling of the test set). **Secure aggregation.** Exactness and dropout are tested with $K = 16$ clients and $t = 9$ on random vectors of the model's dimension, over repeated trials; scaling is measured for $K \in \{4, \dots, 64\}$ and m up to 2^{20}; the statistical indistinguishability of the server view is tested on masked uploads. **Explanation.** For each host we select up to 150 test nodes that are malicious *and* flagged by the detector, i.e. whose score exceeds the threshold that yields a 5% false-positive rate on the benign test nodes (so that there is a detection to explain), with at least eight and at most 60 players, at least three of which are incident to the node, and an excess $v(\mathcal{P}_v) - v(\emptyset) > 10^{-3}$. Nodes whose non-incident edge set duplicates that of an earlier node (neighbouring nodes in the same attack subgraph sample nearly the same ego graph) are skipped, so the evaluation units are not near-duplicates.

7. Results

7.1 Detection Performance

Table 11 and Table 12 report the ranking metrics on the held-out test graphs. Core configurations are averaged over three seeds (plain FedAvg over two, FedProx over one); n is the number of seeds.

Table 11. Detection results on THEIA (test day). Mean ± standard deviation over n seeds. TPR@1% and TPR@5% are true-positive rates at 1% and 5% false-positive rate; P@1000 is the precision among the 1,000 top-ranked nodes; F1* is the oracle best-threshold F1.

Configuration	n	AUROC	AUPRC	TPR@1%	TPR@5%	P@1000	F1*
Raw features (no GNN)	3	0.709 ± 0.275	0.270 ± 0.198	0.094 ± 0.150	0.139 ± 0.134	0.029 ± 0.022	0.396 ± 0.211
Local only (mean of 4 clients)	3	0.987 ± 0.004	0.756 ± 0.048	0.616 ± 0.134	0.997 ± 0.006	0.041 ± 0.034	0.872 ± 0.036
Centralised (no privacy)	3	0.976 ± 0.035	0.732 ± 0.323	0.621 ± 0.541	0.710 ± 0.475	0.321 ± 0.349	0.855 ± 0.160
FedAvg (plain)	2	0.994 ± 0.001	0.850 ± 0.034	0.981 ± 0.016	1.000 ± 0.000	0.026 ± 0.001	0.939 ± 0.018
FedAvg + SecAgg (proposed)	3	0.995 ± 0.001	0.870 ± 0.020	0.988 ± 0.015	1.000 ± 0.000	0.142 ± 0.201	0.952 ± 0.021
FedProx + SecAgg	1	0.995	0.879	0.993	1.000	0.027	0.953

Federation under secure aggregation did not cost ranking quality. On THEIA the proposed configuration attains AUROC 0.995 $\pm$ 0.001 against 0.976 $\pm$ 0.035 for centralised training, whose mean is pulled down by one poor seed (range 0.936 to 0.999); we read this as "no evidence that federation hurts". On CADETS the ordering is consistent across seeds (0.971 $\pm$ 0.007 against 0.934 $\pm$ 0.015); we did not investigate why. Plain and securely aggregated FedAvg are equivalent (0.994 and 0.964), as expected from Proposition 1, and FedProx adds nothing at $\mu = 0.01$ (0.995, 0.963, one seed).

The top of the ranking is not reliable: P@1000 is low and seed-dependent (THEIA 0.142 $\pm$ 0.201, base rate 7.3%); the detector is a filter, not a short list of confirmed entities.

Table 12. Detection results on CADETS (test day). Same protocol as the previous table.

Configuration	n	AUROC	AUPRC	TPR@1%	TPR@5%	P@1000	F1*
Raw features (no GNN)	3	0.259 ± 0.352	0.056 ± 0.035	0.002 ± 0.001	0.004 ± 0.000	0.014 ± 0.001	0.105 ± 0.062
Local only (mean of 4 clients)	3	0.668 ± 0.098	0.286 ± 0.104	0.086 ± 0.144	0.253 ± 0.249	0.030 ± 0.003	0.389 ± 0.111
Centralised (no privacy)	3	0.934 ± 0.015	0.363 ± 0.049	0.003 ± 0.000	0.004 ± 0.000	0.035 ± 0.002	0.533 ± 0.054
FedAvg (plain)	2	0.964 ± 0.003	0.508 ± 0.022	0.004 ± 0.000	1.000 ± 0.000	0.036 ± 0.000	0.675 ± 0.020
FedAvg + SecAgg (proposed)	3	0.971 ± 0.007	0.561 ± 0.053	0.003 ± 0.000	1.000 ± 0.000	0.036 ± 0.001	0.719 ± 0.044
FedProx + SecAgg	1	0.963	0.501	0.003	1.000	0.036	0.668

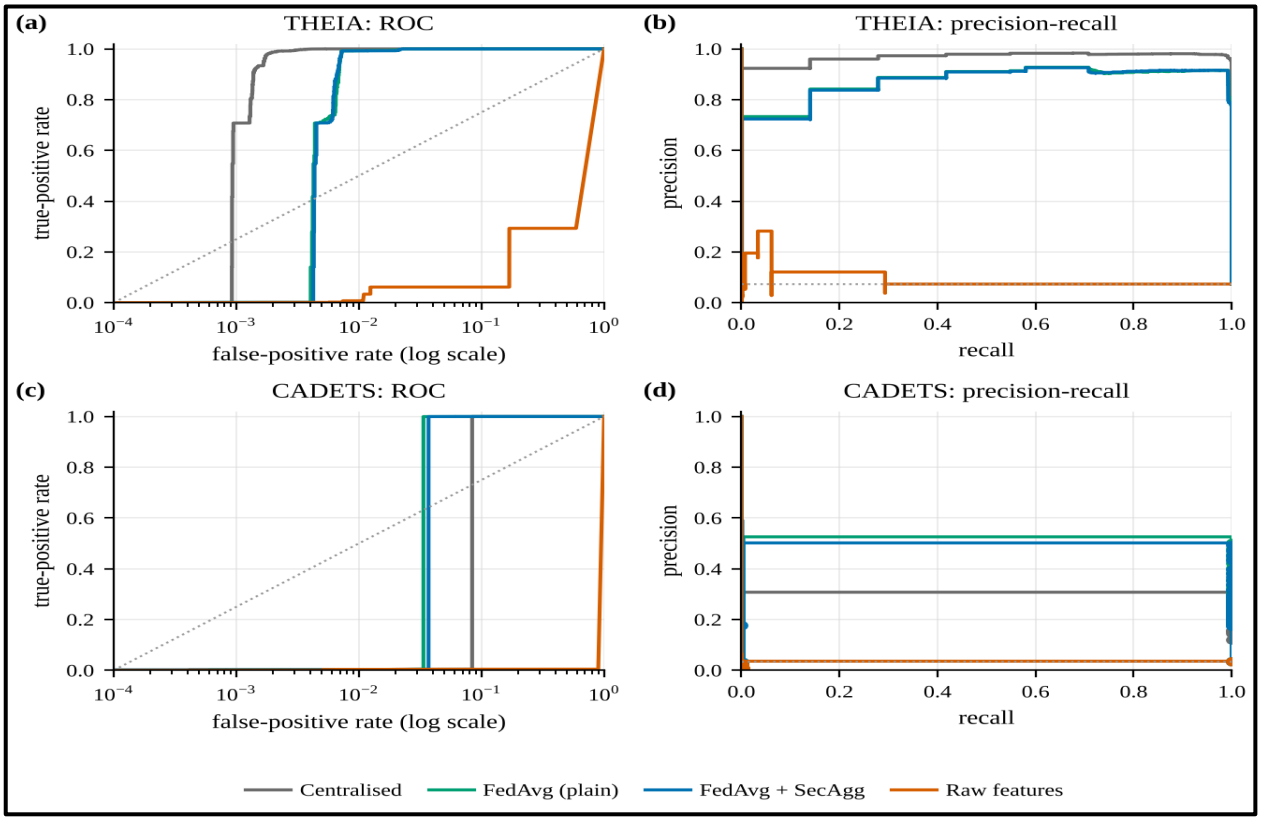


Figure 11. ROC curves (left, logarithmic false-positive axis) and precision-recall curves (right) on THEIA (a, b) and CADETS (c, d) for the configurations of Table 11 and Table 12 (seed 0; local-only models have no single global score and are omitted). The CADETS curves are step functions with a few vertices (markers) because 99.5% of the malicious nodes share a single score.

Federation matters where local data are unrepresentative, and the graph matters everywhere. Local-only models are close in AUROC on THEIA (0.987) but worse at the top of the ranking (TPR@1% 0.616 against 0.988); on CADETS the gap is large (0.668 against 0.971) because two of four clients, having seen one day, rank the attack as *more* normal than benign nodes (AUROC 0.19 and 0.35). The raw-feature baseline is unreliable (AUROC 0.398 to 0.920 on THEIA, 0.056 to 0.665 on CADETS): within the node type that holds the attack, malicious nodes have *lower* degree than benign ones

(degree AUROC 0.18 and 0.33), so a scorer of structural unusualness ranks hubs above the attack leaves. Fig. 12 shows every run.

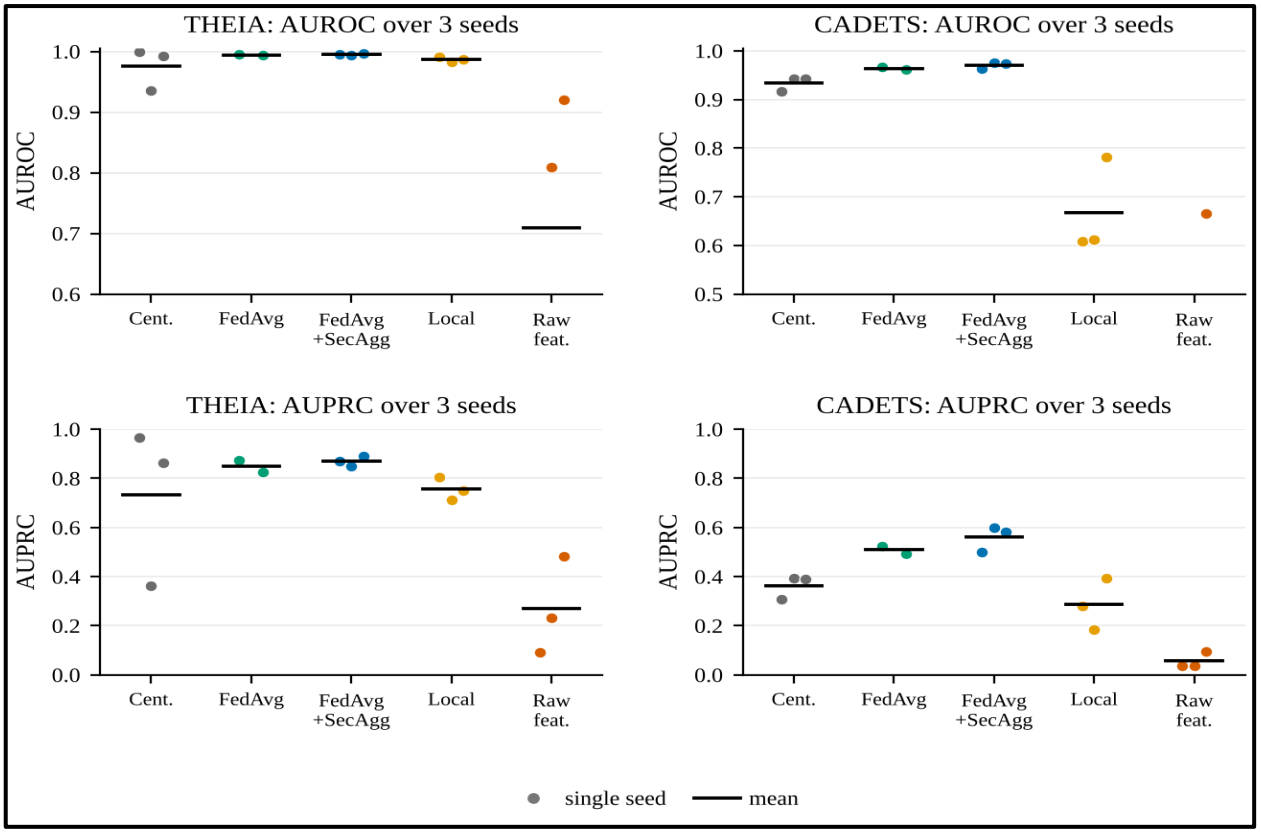


Figure 12. AUROC and AUPRC of every run (dots) and their mean (bar) for the five configurations on THEIA (left) and CADETS (right). Centralised training on THEIA has one outlying seed; local-only and raw-feature models are far below the federated model on CADETS.

7.2 Deployable Operating Points

Table 13 shows what the benign-calibrated thresholds of (17) deliver, with no use of any attack label.

Table 13. Benign-calibrated operating points of FedAvg + SecAgg (mean ± standard deviation over three seeds). The nominal benign alarm rate is 1 − q.

Host	q	Precision	Recall	F1	FPR	Accuracy
THEIA	0.95	0.502 ± 0.063	1.000 ± 0.000	0.667 ± 0.057	0.080 ± 0.021	0.926 ± 0.020
THEIA	0.99	0.805 ± 0.044	0.995 ± 0.008	0.890 ± 0.029	0.019 ± 0.005	0.982 ± 0.005
THEIA	0.999	0.015 ± 0.012	0.000 ± 0.000	0.000 ± 0.000	0.004 ± 0.005	0.923 ± 0.005
CADETS	0.95	0.159 ± 0.186	1.000 ± 0.000	0.247 ± 0.257	0.476 ± 0.358	0.541 ± 0.346
CADETS	0.99	0.248 ± 0.413	0.335 ± 0.575	0.283 ± 0.482	0.013 ± 0.002	0.963 ± 0.020
CADETS	0.999	0.056 ± 0.042	0.002 ± 0.001	0.004 ± 0.002	0.001 ± 0.001	0.963 ± 0.001

On THEIA the 99% point is a usable alarm policy: precision 0.805, recall 0.995, F1 0.890 at a benign false-positive rate of 0.019, above the nominal 0.01, so calibrated alarm rates are lower bounds. The 99.9% point is not usable (recall 0.000) because the attack lies at the 99.6th to 99.8th percentile of the benign scores of its type. On CADETS the operating points are unstable: at 95% every seed recovers the attack but the benign false-positive rate is 6.3% in one seed and about 68% in the other two; at 99% the attack is detected in one seed (recall 0.999, precision 0.725) and missed in the other two (recall 0.003). The mean and deviation in Table 13 therefore summarize a bimodal outcome.

7.3 The CADETS Anomaly: A Detectable Attack at the Margin of the Threshold

The CADETS ROC curve in Fig. 11 rises vertically at a false-positive rate of about 3.7% and is flat afterwards: 99.5% of the 12,846 malicious nodes receive exactly the same score (THEIA: 13.9%), because 99.7% of them are leaves of one type attached through one edge type to a few hubs, so their sampled neighbourhoods and embeddings are identical. The attack is then detected entirely or not at all by any threshold, and where that single score falls in the benign distribution decides which. Fig. 13 shows that it falls at the 99.1st to 99.2nd percentile of the benign nodes of its type. Thresholds are calibrated on benign *training* days, so a drift of a few percent in the score scale moves a threshold across the attack, as the seeds show. Among leaves the separation is nearly perfect in rank terms (leaf-only AUROC 0.995, THEIA 0.999): the detector *scores* the attack well, the *threshold* is brittle.

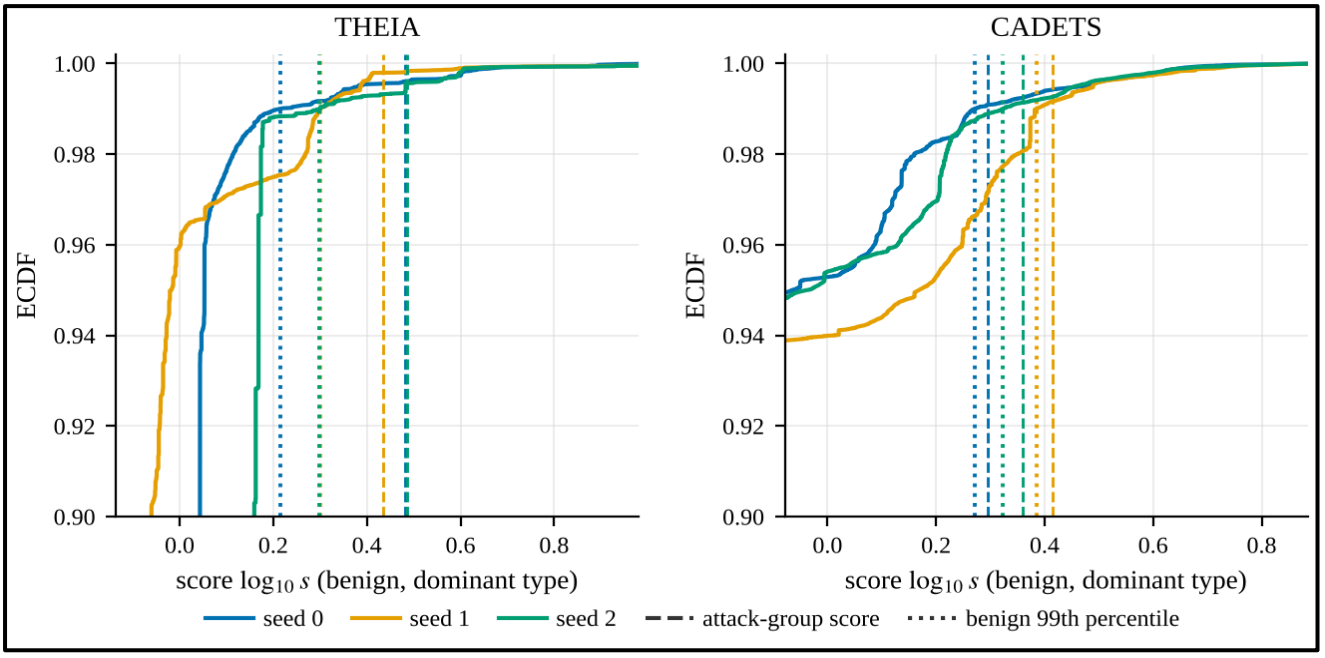


Figure 13. Empirical CDF of the benign scores of the dominant node type for each seed (solid), with the score of the attack group (dashed) and the 99th percentile of the benign test scores (dotted); the vertical axis starts at 0.90. On THEIA the attack lies well above the benign bulk; on CADETS it lies within a percentile of the 99% line.

7.4 Threat Detection by Attack Type

AUROC over all nodes does not say *which part of the campaign* is found. Table 14 breaks detection down by the entity classes of Table 6: scanning activity (A and B), the exploited application, loader and command-and-control infrastructure (C), and the implant process (hub); Fig. 14 plots the same breakdown with per-seed recall.

Table 14. Detection by attack type (FedAvg + SecAgg, mean over three seeds). Recall is the fraction of the class flagged at the benign-calibrated thresholds $q = 0.95/0.99/0.999$; for the hub, flagged seeds of three. The class-to-behaviour mapping is the author interpretation of Table 6.

Attack behaviour (ATT&CK)	Class	THEIA AUROC	THEIA recall	CADETS AUROC	CADETS recall
Network scanning (T1046)	A	0.996	1.00 / 1.00 / 0.00	0.971	1.00 / 0.33 / 0.00
Scan targets, staging (T1046, T1105)	B	0.994	1.00 / 0.98 / 0.00	0.985	1.00 / 0.78 / 0.44
Exploit, loader, C2 channel (T1190/T1189, T1105, T1071)	C	0.989	0.92 / 0.67 / 0.14	0.992	0.85 / 0.71 / 0.40
Implant process (T1055, T1071)	Hub	99.95th pct.	3/3 / 2/3 / 1/3	99.67th pct.	3/3 / 2/3 / 1/3

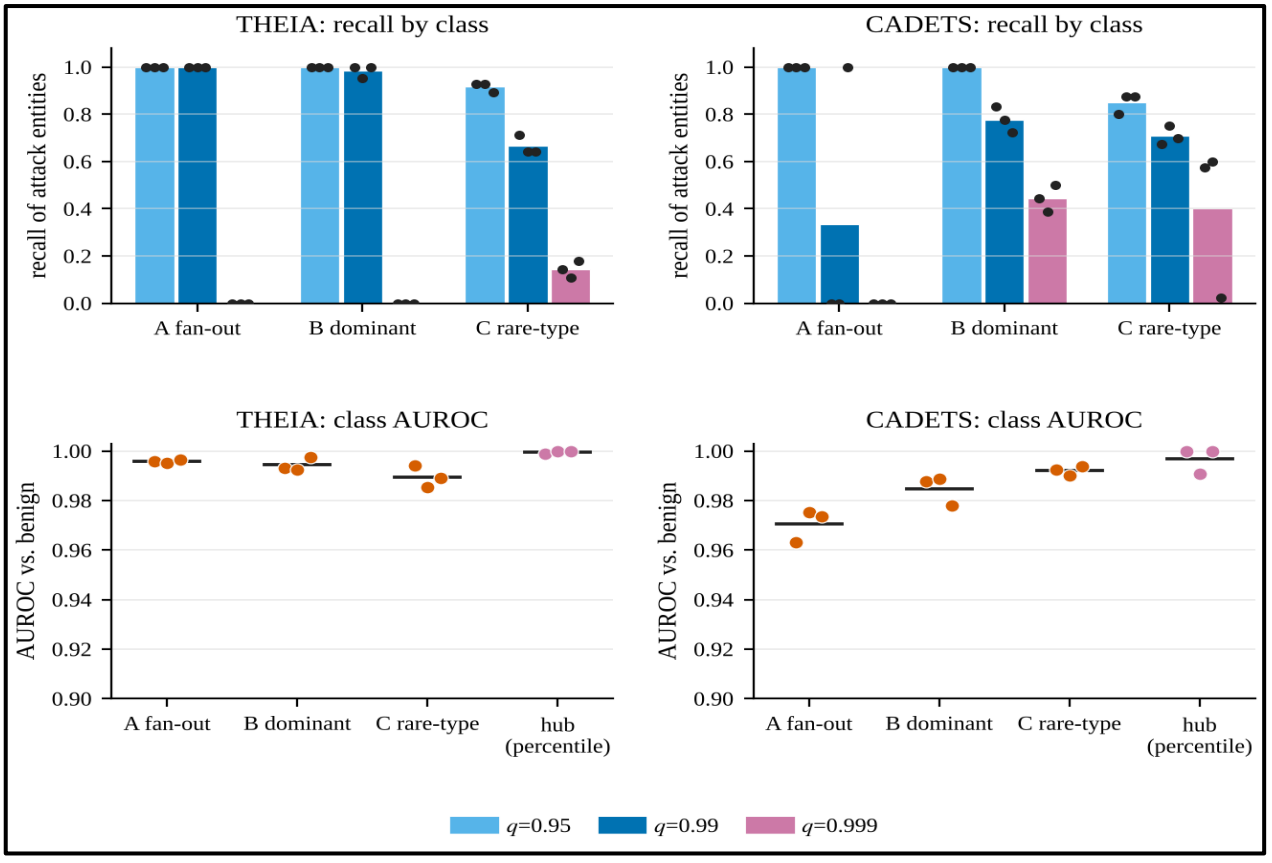


Figure 14. Detection by attack-entity class. Top: recall at the three calibrated thresholds (bars: mean; dots: seeds). Bottom: AUROC of each class against the benign nodes, and the hub's score percentile among the benign nodes. Scan-like fan-out leaves (A) are fully recovered at $q = 0.95$; on CADETS they are recovered at $q = 0.99$ in one seed of three.

(i) Every class is ranked far above benign activity: class AUROC is 0.996 (THEIA) and 0.971 (CADETS) for the fan-out leaves and 0.989 and 0.992 for the rare-type entities, and the hub lies at the 99.95th and 99.67th percentile of the benign scores. (ii) Scan activity is found completely at the 95% threshold on both hosts but not reliably at 99% on CADETS, where the shared score sits at the margin of the threshold (Fig. 13). (iii) Exploit and C2 entities (class C) are the hardest to alarm on: at $q = 0.99$ only 0.64 to 0.71 (THEIA) and 0.68 to 0.75 (CADETS) are flagged, and at $q = 0.999$ 0.14 and 0.40. These few entities are the most informative for an analyst but are diluted by thousands of scan leaves in aggregate metrics. The labels carry no scenario or stage tags, so recall per scenario or per kill-chain stage cannot be computed; the class breakdown is the finest resolution the data permit.

7.4.1 Alert Workload

A hub with 10^4 leaves produces 10^4 flagged nodes, so alerts are grouped by their *anchor*, the highest-degree neighbour of the flagged node. Table 15 and Fig. 15 give the effect.

At $q = 0.99$ the 31,345 flagged THEIA nodes collapse into 913 groups, 10.7 of which contain attack entities (precision about one percent); on CADETS the figures are 1323 and 8.3. The attack is compact and always inside the alert list when the threshold is crossed, but an analyst would still triage several hundred benign groups. Fig. 16 shows the non-leaf attack entities.

Table 15. Alert workload of the proposed model (mean ± standard deviation over three seeds). Anchor groups collapse flagged nodes that share their highest-degree neighbour; group precision is the share of groups that contain at least one attack entity; coverage is the share of attack entities that are flagged.

Host	q	Flagged nodes	Anchor groups	Groups with attack	Group precision	Coverage
THEIA	0.95	50978 ± 6769	2045 ± 412	11.3 ± 0.6	0.6 ± 0.1%	100.0 ± 0.0%
THEIA	0.99	31345 ± 1483	913 ± 253	10.7 ± 0.6	1.2 ± 0.3%	99.5 ± 0.8%
THEIA	0.999	1264 ± 1721	148 ± 28	5.3 ± 0.6	3.6 ± 0.3%	0.0 ± 0.0%
CADETS	0.95	176878 ± 123410	3301 ± 112	9.0 ± 0.0	0.3 ± 0.0%	100.0 ± 0.0%
CADETS	0.99	8816 ± 7708	1323 ± 500	8.3 ± 0.6	0.7 ± 0.4%	33.5 ± 57.5%
CADETS	0.999	492 ± 227	117 ± 2	5.7 ± 1.2	4.8 ± 0.9%	0.2 ± 0.1%

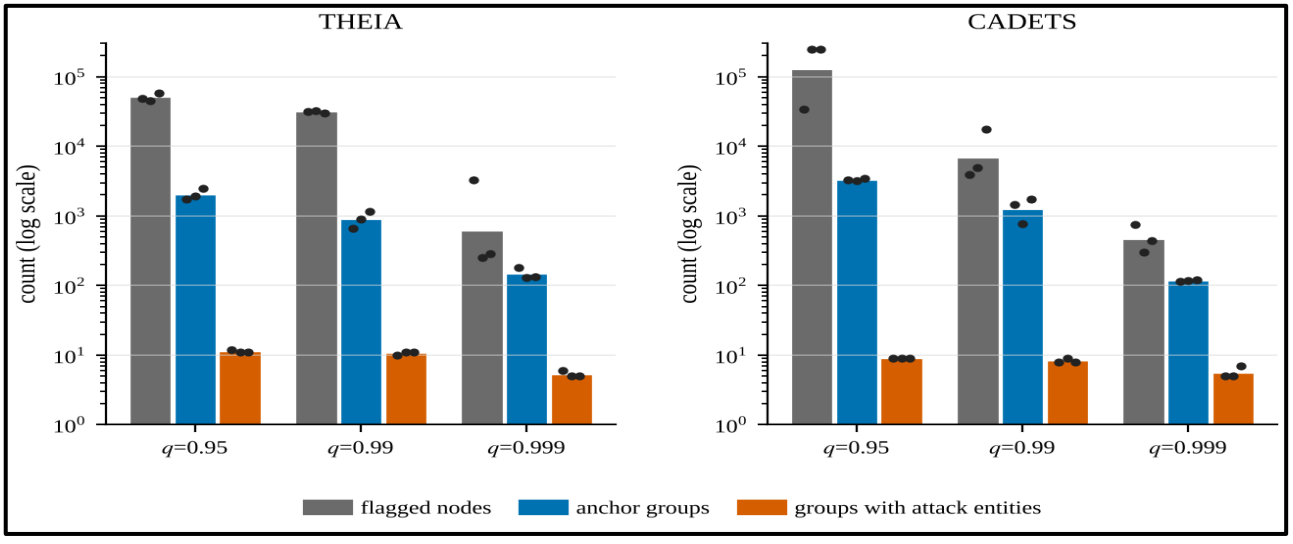


Figure 15. Alert workload at the three calibrated thresholds: flagged nodes, anchor groups, and groups containing attack entities (geometric mean over seeds, dots are seeds). Grouping reduces the alerts by a factor of about 4 to 50, but most groups at $q \leq 0.99$ remain benign.

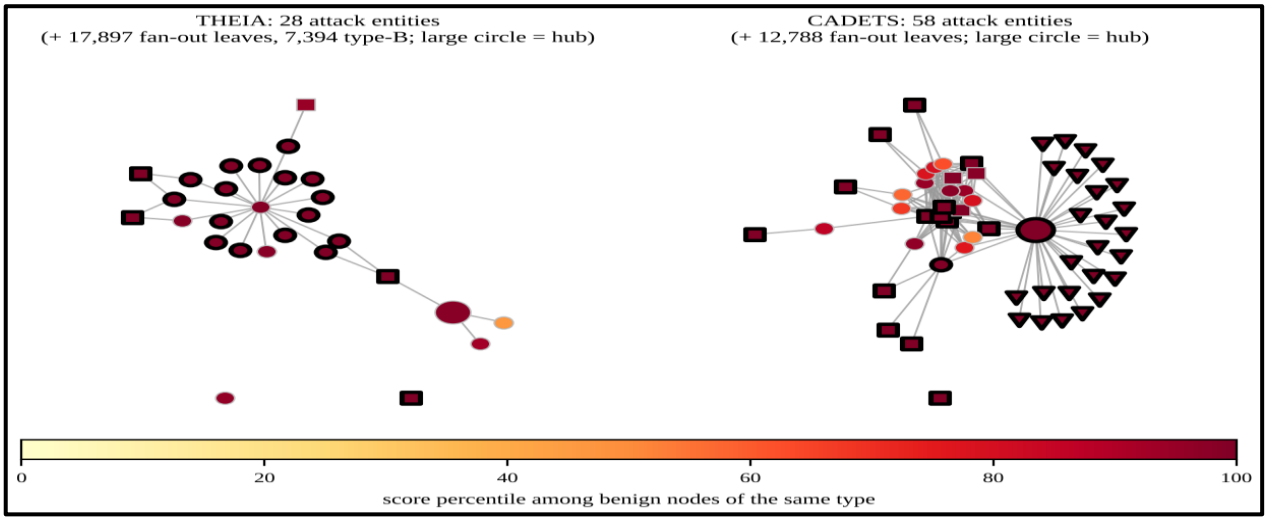


Figure 16. Induced subgraph of the labelled attack entities that are not fan-out leaves (THEIA: rare-type entities only; CADETS: all 58). Colour is the score percentile among benign nodes of the same type, marker shape the node type, and a black ring marks a node flagged at $q = 0.99$ (seed 0). The large circle is the hub; the fan-out leaves are omitted for legibility.

7.5 Ablation of Model Components

Table 16 removes one component at a time (single seed unless stated; the last column is the AUROC change against the full model's three-seed mean).

Table 16. Ablation of the model components under FedAvg + SecAgg.

Host	Variant	n	AUROC	AUPRC	F1*	Δ AUROC
THEIA	Full model (edge-typed GATv2, masked)	3	0.995 ± 0.001	0.870 ± 0.020	0.952 ± 0.021	—
THEIA	Uniform aggregation (GraphSAGE-style)	1	0.991	0.797	0.882	-0.005
THEIA	No edge-type information	1	0.992	0.811	0.942	-0.003
THEIA	No masking of the seed features	1	0.997	0.883	0.978	+0.001
THEIA	No graph (MLP on own features)	1	0.668	0.245	0.419	-0.327
CADETS	Full model (edge-typed GATv2, masked)	3	0.971 ± 0.007	0.561 ± 0.053	0.719 ± 0.044	—
CADETS	Uniform aggregation (GraphSAGE-style)	1	0.948	0.414	0.587	-0.023
CADETS	No edge-type information	1	0.985	0.711	0.833	+0.014
CADETS	No masking of the seed features	1	0.958	0.466	0.638	-0.013
CADETS	No graph (MLP on own features)	1	0.668	0.097	0.178	-0.302

The graph is essential: an MLP on the node's own features lowers AUROC to 0.668 (THEIA) and 0.668 (CADETS), far outside the seed deviation. Learned attention is probably useful: uniform aggregation lowers AUPRC from 0.870 to 0.797 and from 0.561 to 0.414, each from a single run. Edge-type embeddings did not help: removing them changes AUROC by -0.003 on THEIA and *improves* it by 0.014 on CADETS, plausibly because edge types already enter through the histogram features. Masking the seed features has no consistent effect. We keep edge-typed attention and masking for reasons of method (the explainer of §5.8 uses per-relation messages), not because the ablation demands them.

7.6 Cost of Federation

The seed-to-seed deviation of the proposed model is 0.001 AUROC on THEIA and 0.007 on CADETS, so the larger effects above (federated versus local, graph versus no graph) are meaningful and the smaller ones (plain versus secure aggregation, FedProx) are not. All cross-client sums of a run (45 calls) take 0.1 s of client and 0.0 s of server computation on THEIA, 0.1% of the run time (CADETS: 0.1%), with a per-client uplink of 15.3 MB (THEIA) and 16.0 MB (CADETS) for the whole pipeline. When every client drops out of each model-update round with probability 25%, the protocol completes 21 of 30 rounds on THEIA and 21 on CADETS (a round fails when two of four clients drop out, probability 26%); the final AUROC is 0.997 and 0.957 against 0.995 and 0.963 without dropout, a change of the order of the seed spread.

7.7 Secure Aggregation: Exactness, Dropout, Scaling and Leakage

This section tests the protocol of §5.6 on its own, using the implementation that also drives the training runs, with real cryptographic operations (X25519, HKDF, AES-256-GCM transport of Shamir shares, ChaCha20 mask expansion).

7.7.1 Exactness, Dropout and Scaling

We aggregate vectors of the size of the model update ($m = 37{,}778$) among $n = 16$ clients with threshold $t = 9$, with clients vanishing before uploading their masked vector or after uploading it, ten trials per scenario (Fig. 17a). Every scenario in which at least t clients survive the upload round returns the sum of exactly the surviving clients, with a maximum error of 3.2e-07, within the bound $|U_3| \cdot 2^{-s-1}$ of Proposition 1 (4.8e-07 for $n = 16$, $s = 24$). With fewer than t survivors the call fails in every trial (0/10 successful) and no partial sum is produced: the protocol trades availability for the guarantee that a sum of fewer than t clients is never revealed.

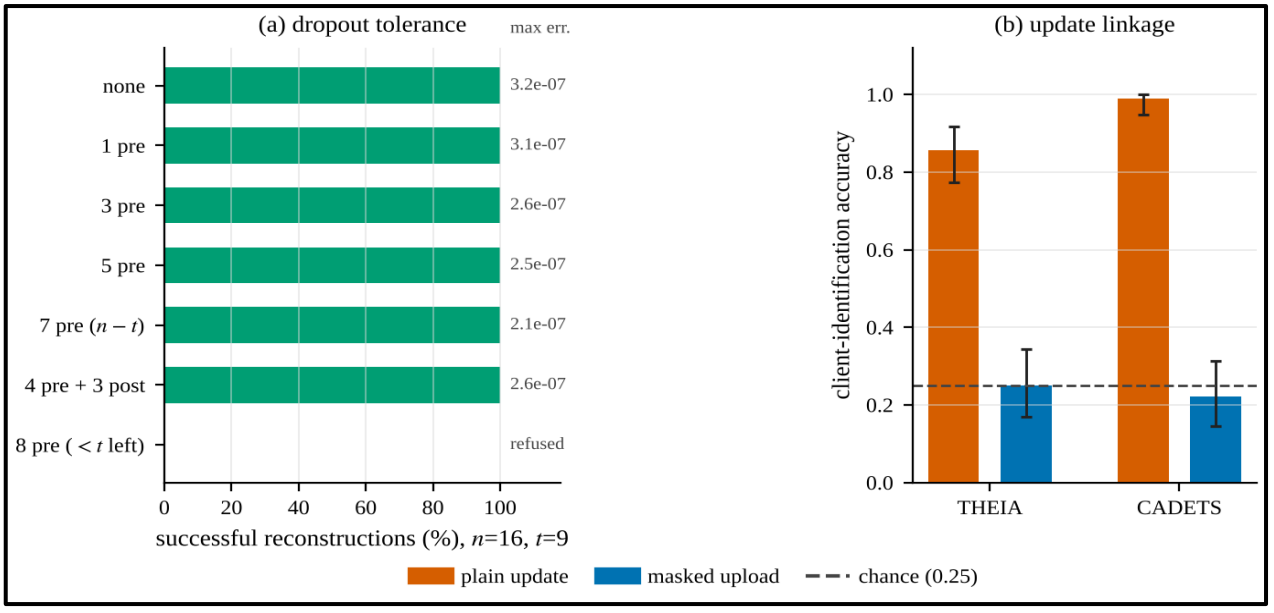


Figure 17. Robustness and privacy of the secure layer. (a) Successful reconstructions out of ten trials for six dropout patterns with at least $t = 9$ survivors and one with fewer, with the maximum absolute error beside each bar. (b) Accuracy of the linkage attack on plain updates and on masked uploads against chance, with 95% confidence intervals.

Fig. 18 shows the cost versus n and m on one CPU core. For the model of this paper and $n = 16$, one call costs a client 5.6 ms and 306 KB of uplink (2.02 times the float32 size) and the server 5.4 ms. The ring element is 8 bytes against 4 for float32, so the vector alone doubles the payload, and the per-client overhead (keys, n encrypted shares, unmasking reply) is a few kilobytes growing linearly in n. Client and server time grow roughly linearly in n and m (22.2 ms against 54.7 ms at $n = 64$; we did not profile the faster-than-linear growth of the server time), and a call stays well below a second for 32 clients and a million-entry update.

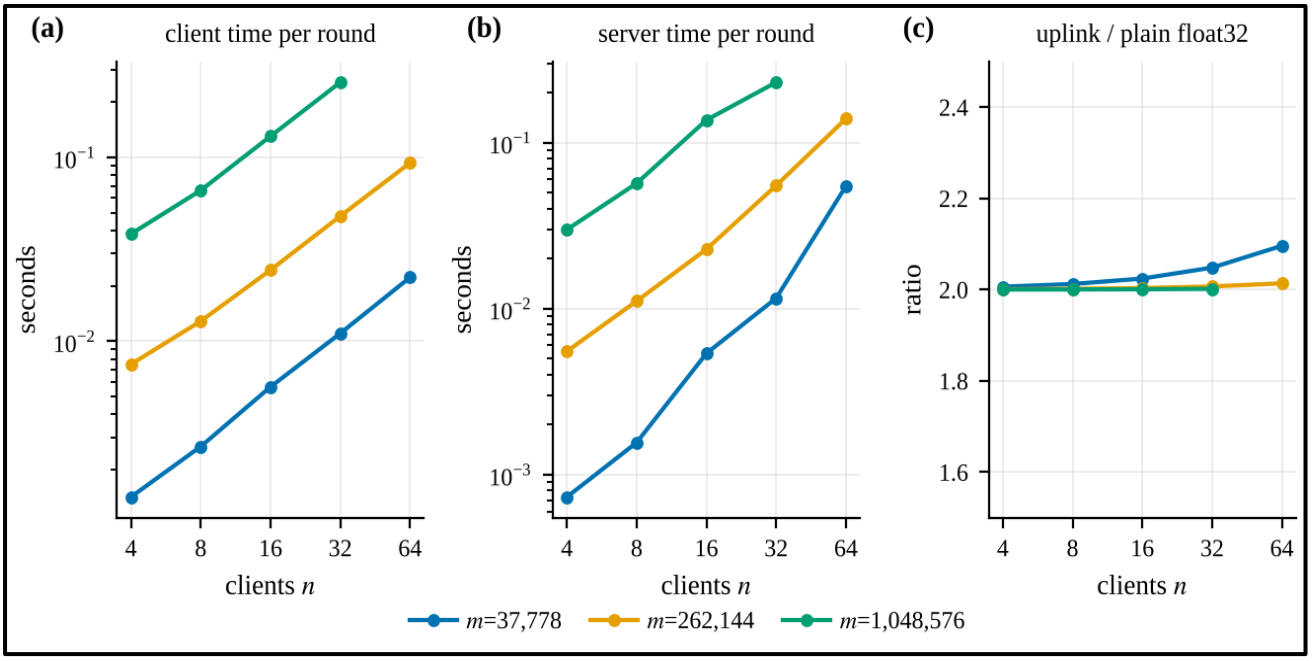


Figure 18. Cost of one secure-sum call versus the number of clients for three vector lengths m: client time, server time (both logarithmic) and the uplink ratio to the float32 vector. Time grows linearly in n for the client and the payload ratio stays at about 2.

7.7.2 What the Server Sees

In each of 60 trials, 8 clients upload $m = 20{,}000$ masked entries and we test one upload for (i) uniformity of the low byte (χ^2 test) and (ii) linear correlation with the client's plain values. The mean p-value is 0.522 with 0.95 of trials above 0.05 (Kolmogorov-Smirnov of the p-values against uniform: $p = 0.949$), and the correlation with the plain values has mean -0.00023 and maximum absolute value 0.0201. These tests cannot prove security, but they would detect implementation errors such as mask re-use or a truncated mask.

7.7.3 Linkage Attack across Rounds

A server that sees plain updates can try to follow a client across rounds because updates of one client resemble each other. For each pair of consecutive rounds, the server receives the $K = 4$ vectors of round $r + 1$ in random order and assigns each to the client of round r with the largest cosine similarity; chance is 25%. We run it on the saved plain updates of the FedAvg runs (seed 0), on the same updates with the cross-client mean removed, and on the masked uploads under the protocol (Fig. 17b).

Plain updates are linkable: the attack identifies the client of 85.6% (THEIA) and 99.0% (CADETS) of the 104 anonymized updates, against 25% by chance, and subtracting the round mean raises the THEIA rate to 89.4%. Each client's update has a persistent signature, which is not surprising, since each client trains on its own day of data from a common starting point and with its own optimiser trajectory. The masked uploads show no such signature: the accuracies are 25.0% and 30.8% (THEIA) and 22.1% and 18.3% (CADETS), and none of the four differs significantly from chance (smallest two-sided $p = 0.14$). This is the behaviour that §5.10 predicts, because the server's view of a masked upload is independent of the plain update without the mask seeds. The test is narrow: it covers one attack (nearest neighbour in cosine similarity) on $K = 4$ clients over 104 trials per condition, so a failure of this attack is not a proof that no other inference is possible, and the *aggregate* is deliberately not protected by this mechanism (§8.5).

7.8 Edge-Level Shapley Explanations

We evaluate the explainer of §5.8 on detections of the proposed model (seed 0). On THEIA, 150 distinct flagged malicious nodes satisfy the selection rules of §6.5 (on average 26.8 players per ego graph, 3.7 incident to the target). On CADETS only 19 distinct nodes remain, because the malicious CADETS nodes are leaves of a handful of hubs and share ego graphs; those results rest on a small sample.

7.8.1 Efficiency and Fidelity

The largest violation of Proposition 2 over all explained nodes is 1.9e-05 (THEIA) and 9.4e-06 (CADETS), i.e. single-precision rounding. Table 17 and Fig. 19 report deletion and insertion fidelity of (22) (medians, because the denominators are occasionally near zero).

Table 17. Deletion and insertion fidelity (median over explained nodes) and top-ranked-edge statistics. D(k): fraction of the anomalous excess removed by replacing the top-k edges with benign-typical messages; I(k): fraction recovered by keeping only the top-k edges.

Host	Method	D(1)	D(3)	I(1)	I(3)	Nodes with D(3) ≥ 0.5	Top-1 edge incident
THEIA	Random	0.01	0.02	0.00	0.00	23%	12%
THEIA	Attention	0.00	0.02	0.00	0.00	1%	1%
THEIA	Saliency	0.57	1.00	0.23	0.94	97%	100%
THEIA	GNNExplainer-lite	0.58	1.00	0.13	0.94	92%	85%
THEIA	Occlusion	0.61	1.00	0.16	0.94	97%	99%
THEIA	Edge-Shapley	0.57	1.00	0.23	0.94	98%	99%
CADETS	Random	0.00	0.02	0.00	0.05	0%	42%
CADETS	Attention	0.01	0.01	0.00	0.00	26%	11%
CADETS	Saliency	0.25	0.91	0.85	0.98	79%	95%
CADETS	GNNExplainer-lite	0.11	0.63	0.45	0.87	58%	95%
CADETS	Occlusion	0.54	0.91	0.52	0.97	74%	100%
CADETS	Edge-Shapley	0.25	0.82	0.75	1.12	68%	100%

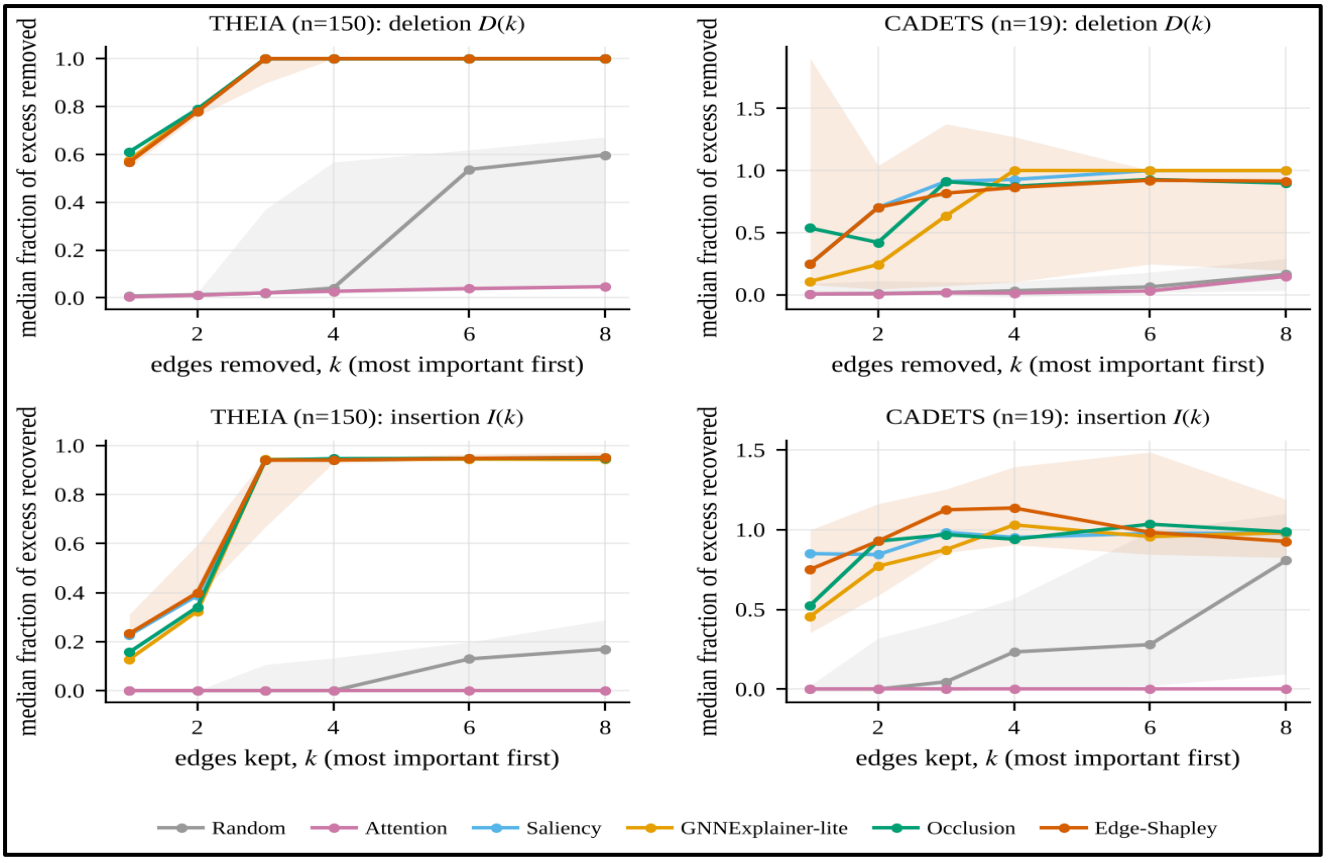


Figure 19. Deletion (top) and insertion (bottom) curves of the six attribution methods on THEIA (left, 150 nodes) and CADETS (right, 19 nodes): median and, for Shapley and random, the inter-quartile band. Attention and random rankings barely move the score; the four model-based methods are nearly indistinguishable beyond $k = 3$.

First, attention is *not* an explanation of this model: replacing the three most-attended edges removes a median 0.02 of the excess, no better than random edges (0.02), and attention mass is lowest on the edges incident to the target (AUROC 0.01), in line. Second, the four model-based methods are close: the top three edges remove the whole excess (median D(3) = 1.00 for Shapley on THEIA) and keeping only them recovers 0.94. Paired over nodes, Shapley differs from occlusion by -0.002 in deletion (bootstrap CI [-0.003, -0.001]): detectable but practically negligible, so on THEIA Shapley *matches* the best cheap baseline rather than beating it; the CADETS medians are noisier (19 nodes). Third, the metric cannot separate the methods because 94.0% of the absolute Shapley mass (CADETS 89.6%) lies on the target's own incident edges, which every model-based method finds. Shapley's advantages are its axioms: values are *complete*, *signed* (88.2% of THEIA values are positive) and quantify each edge's share of the explanation rather than only an ordering.

7.8.2 Alignment with Ground-Truth Labels

Fig. 20 tests whether the attribution ranks edges that analysts labelled as malicious above the rest. All model-based methods rank the target's own edges first (AUROC 1.00 for Shapley on THEIA). Against the *any-malicious* and *both-malicious* labels Shapley reaches only 0.41 and 0.21 on THEIA, below chance, because the target sits inside a dense malicious neighbourhood (65.3% of ego-graph edges have a malicious endpoint) and the ranking concentrates mass on the target's few incident edges. Among incident edges, Shapley separates edges to malicious neighbours with AUROC 0.79, ahead of occlusion (0.32) and saliency (0.79). On CADETS the alignment is higher (any-malicious 0.83), with 11 to 19 nodes per cell. Entity-level labels are a proxy: an edge between a malicious process and a benign library can be exactly the evidence that makes the process anomalous, so we regard the low alignment as a limitation of the evaluation as much as of the explainer.

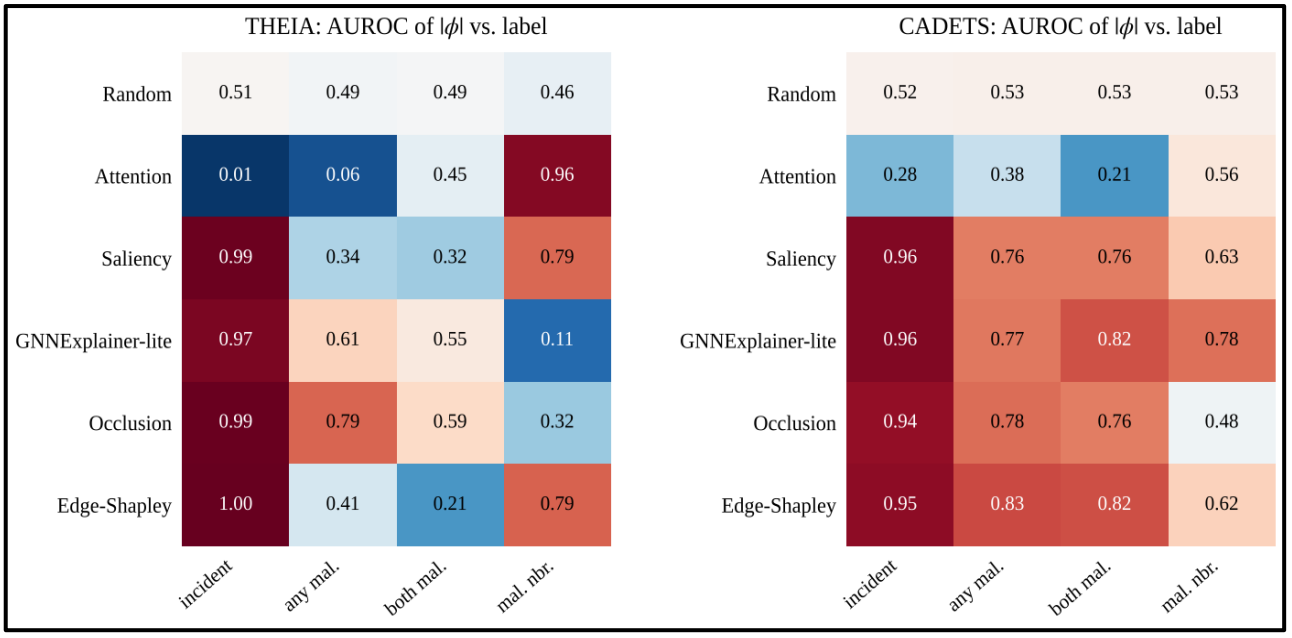


Fig. 20. Ground-truth alignment of the six methods: AUROC of $|\phi_e|$ against four edge-label sets (columns), THEIA (left) and CADETS (right); 0.5 is chance. Values below 0.5 mean that the method ranks label-positive edges lower.

7.8.3 Convergence, Cost and a Case Study

Fig. 21 compares the estimator with an $M = 800$ reference on 30 nodes per host. Rank agreement grows from Spearman 0.59 at $M = 5$ to 0.94 at $M = 200$ on THEIA (CADETS 0.81 to 0.96), whereas the top-three edges an analyst reads are identical to the reference for almost every node from $M = 5$ on THEIA (CADETS 0.84 at $M = 25$). The efficiency gap stays at rounding level for every M. One attribution costs 0.91 s on THEIA (3.63 s on CADETS) on one CPU core, two to three orders of magnitude more than saliency or occlusion; it is run only for flagged nodes and parallelises over permutations.

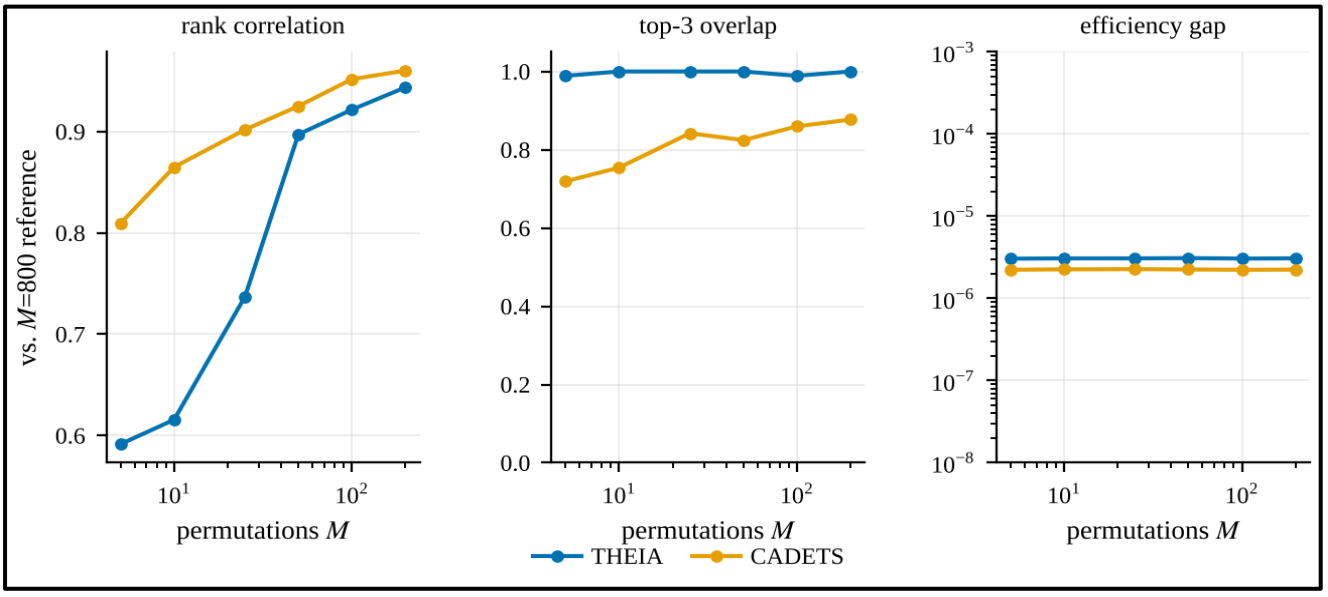


Figure 21. Convergence of the Shapley estimator with the number of permutations M: rank correlation with the $M = 800$ reference (left), overlap of the three highest-ranked edges (centre) and the efficiency gap (right, logarithmic), which is at rounding level for every M.

Fig. 22 shows one explained THEIA detection: a malicious entity with 3 incident edges among 27 players. Its excess is $\nu(\mathcal{P}_v) - \nu(\emptyset) = 1.94$ (score 2.42 against 0.48), the Shapley values sum to 1.94, and the three incident edges carry 95% of the absolute mass with nearly equal attributions. The analyst sees that the score is explained by the node's three direct relationships and that none alone is responsible.

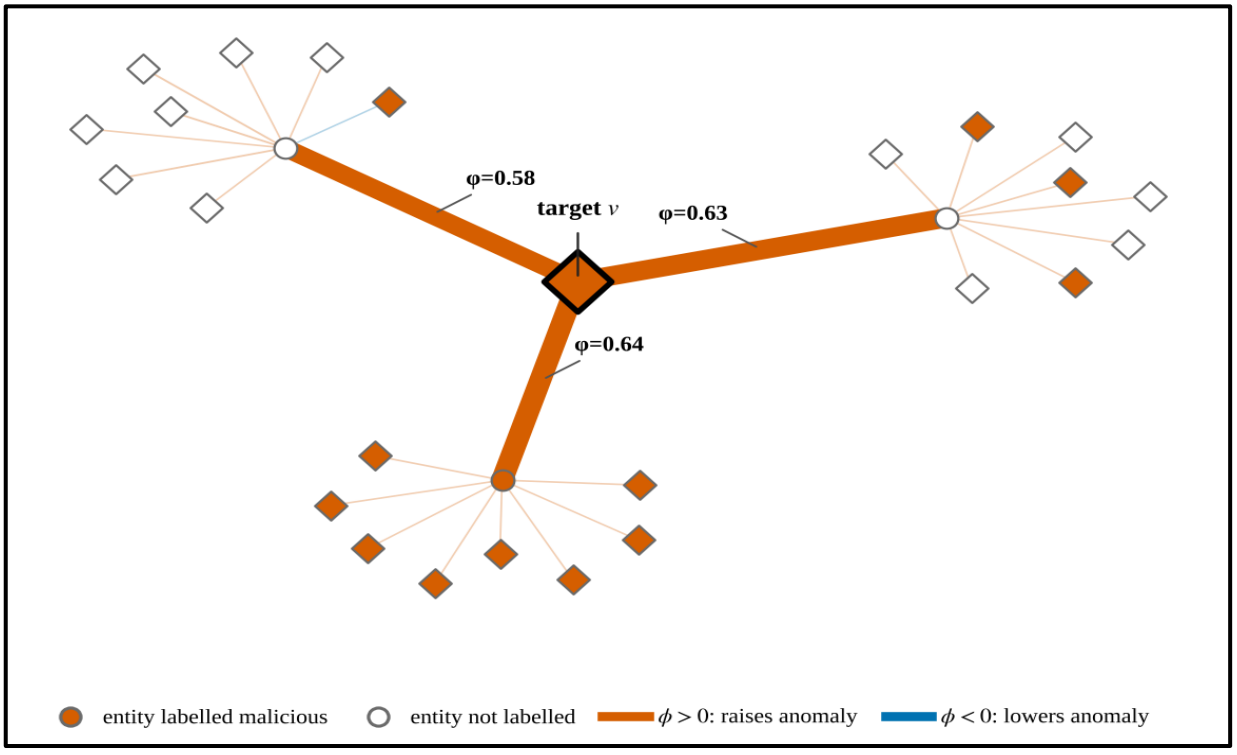


Figure 22. Case study on THEIA: Shapley attribution of one flagged malicious entity over its sampled two-hop provenance neighbourhood. Edge width is proportional to $|\phi|$; red edges raise the anomaly score. The labelled values are the three largest attributions; marker shape is the node type.

8. Discussion

8.1 Summary of Findings

Table 18 condenses the evidence for the three problems of §1.2. Removing the graph is the most damaging ablation; learned attention probably helps, whereas edge-type embedding's and feature masking show no consistent benefit; plain and secure FedAvg are indistinguishable in quality; and the Shapley attribution matches rather than beats occlusion and saliency.

8.2 Interpretation

Why does the federated model not lose to the centralized one? The two share the architecture, the number of gradient steps and the data. The differences are the averaging of four client models every 25 steps and the resetting of the optimizer state. In our runs the centralized model is more variable across seeds (AUROC standard deviation 0.035 on THEIA against 0.001), which is compatible with the view that averaging damps the noise of a single optimizer trajectory in a self-supervised objective that has no label to anchor it. The robust conclusion is the weaker one: federation under secure aggregation did not cost detection quality here.

Why is detection brittle on CADETS although the AUROC is high? Because the attack is a population of near-identical nodes. When 99.5% of the attack nodes share one score, the ranking metrics treat them as one point of the ROC curve, and what matters operationally is on which side of that point a threshold falls. The attack sits at about the 99.1st percentile of the benign scores of its type, within a percentile of the 99% threshold the protocol uses. That the AUROC is nevertheless high (0.971) is not a contradiction: it summarizes the whole ranking, in which the attack outscores between 96.3% and 97.5% of all benign nodes, depending on the seed.

Why does the explainer not separate from simple baselines? Because the metric is saturated by a property of the data. The anomaly of a node is driven by its own incident edges, and the sampled two-hop context adds little. A method must

rank those few edges first to score well, and four of the six methods do. Shapley offers properties that fidelity does not measure (completeness, sign, an inspectable baseline); whether they justify a cost two to three orders of magnitude higher depends on whether an analyst needs to say *how much* an edge contributed or only *which* edges to look at.

8.3 Positioning Relative to Prior Work

Provenance-based detectors have been evaluated on DARPA TC under different unit-of-detection conventions (events, entities, time windows, whole graphs) and different label sets, and published figures are therefore not comparable with ours without re-running the methods under one protocol **[31], [32]**. We do not quote figures from these papers for that reason.

Table 18. Summary of findings per problem. Values are means over three seeds unless noted; the last column states the conclusion the evidence supports.

P	Measure	THEIA	CADETS	Reading
P1	AUROC / AUPRC, FedAvg + SecAgg	0.995 / 0.870	0.971 / 0.561	Attack ranked above benign without labels
P1	AUROC, centralised reference	0.976	0.934	Federation under SecAgg did not cost quality
P1	AUROC, local-only	0.987	0.668	Federation gain is small on THEIA, large on CADETS
P1	F1 at the 99% operating point	0.890	0.283	Usable on THEIA; bimodal across seeds on CADETS
P1	Scan fan-out (A), recall at $q = 0.99$	1.00	0.33	Found completely on THEIA; one seed in three on CADETS
P1	Exploit/C2 entities (C), recall at $q = 0.99$	0.67	0.71	Ranked high, only partly alarmed
P2	Largest relative error of the secure sum	1.19e-07	1.19e-07	Exact to the fixed-point bound
P2	Share of training time spent in SecAgg	0.1%	0.1%	Negligible at this model size
P2	Linkage accuracy, plain update → masked	85.6% → 25.0%	99.0% → 22.1%	Masked uploads at chance (25%)
P3	Largest efficiency violation	1.9e-05	9.4e-06	Exactly complete
P3	Attribution mass on edges incident to the target	94.0%	89.6%	Explains why cheap baselines match Shapley
P3	AUROC vs. any-malicious edge label	0.41	0.83	Weak alignment beyond incident edges
P3	Time per explained node	0.91 s	3.63 s	2–3 orders above occlusion or saliency

8.4 Deployment Considerations

Communication. A full update of the encoder-decoder is 37,778 parameters (about 150 KB in float32) and the 64-bit ring doubles the payload on the wire. With the protocol overhead measured in §7.7, one aggregation round costs each client roughly 306 KB of uplink for $K = 16$ clients (2.02 times the float32 size) and the full pipeline of statistics, 30 training rounds, k-means and calibration costs each client 15.3 MB (THEIA) to 16.0 MB (CADETS) in total. This is negligible next to the audit stream a host produces in a day, which is what is *not* transmitted.

Who computes what. Detection and explanation run at the data owner on its own graph using only the shared model, prototypes and thresholds, so no embedding or score leaves it; the interventional reference messages are one more securely aggregated mean.

Alert triage. An operating point is useful only if the benign test distribution matches the calibration distribution; on CADETS the nominal 1% alarm rate holds on benign nodes but the attack score lies under the threshold (§7.3). A deployment should therefore monitor alarm volume per node type and re-calibrate periodically, again with a securely aggregated histogram.

8.5 Limitations and Threats to Validity

Simulated federation. The four clients are the day-graphs of one host. Inter-organisation heterogeneity (different operating systems, software stacks, workloads, sensors) is not represented, and the federated-versus-local gap measured here may therefore under- or overstate what a real cross-organisation deployment would show. Likewise $K = 4$ is far smaller than a realistic cross-silo or cross-device deployment, and with so few participants the information in the sum itself is substantial (§5.10).

Dataset. DARPA TC Engagement 3 is a 2018 red-team exercise on three hosts. It is the community's standard benchmark, which makes our numbers comparable in spirit to the literature, but its ground truth is entity-level, its attacks are scripted, it contains only one attack day per host, and recent analyses question how well results on it transfer to deployments. We evaluated on two of the three hosts because TRACE fails the independence audit (§4.5). Our structural features (type and degree histograms) do not use node names, command lines or file paths, so the detector cannot overfit to artefacts such as attacker-chosen file names; but it may also be missing signal that semantic features would provide.

Label semantics. Labels mark entities that took part in the attack, not malicious *events*; a node flagged because it is anomalous in context but absent from the list counts as a false positive, so absolute precision should be read accordingly.

Test-set exposure. The prototype scorer and $K = 256$ were chosen in exploratory runs that evaluated AUROC on the test graphs, so absolute numbers may be mildly optimistic; comparisons between configurations are unaffected because hyperparameters are shared.

Security model. Only semi-honest security is provided (§5.10). Poisoning, model inconsistency, and evasion are not evaluated. Provenance detectors are known to be vulnerable to mimicry and adversarial system actions; a model trained to recognise "unusual context" is exactly the kind of model such attacks target, and we do not claim robustness. The linkage attack of §7.7 is a *specific* privacy test, not a proof that no inference is possible from the aggregate.

Explanation validity. The Shapley values explain *the model's score* under an interventional value function that replaces edge messages with benign-type means. They do not establish causal facts about the attack, they depend on the sampled ego graph (capped at 60 players), and a different baseline would give different values. Alignment with entity-level labels is a proxy, and CADETS is evaluated on only 19 distinct nodes.

Statistical power. The core configurations use three seeds and the other ablations one; differences of the order of the seed standard deviation should not be interpreted.

8.6 Future Work

Several extensions follow directly from the limitations. A multi-host or multi-organization federation, in which each client is a different host with its own operating system and workload, would test whether the encoder generalizes across heterogeneous clients and would make the personalized-FL literature **[41], [42]** relevant. The protocol should be hardened to the malicious-server setting with the signature-based consistency round of the full protocol, extended with differential privacy, and made scalable with the sparse-graph construction of **[20]**. Richer node features, for instance semantic embedding’s of paths and command lines, should improve discrimination of leaf nodes, which is the main weakness exposed on CADETS. The Shapley explainer could be extended to *paths* rather than edges, which matches the way analysts’ reason about provenance, and amortized with a learned explainer **[23]** to reduce the cost per alert. Finally, evaluation on newer and larger enterprise-scale corpora, and under adaptive attackers, is needed before any deployment claim.

9. Conclusion

We built and evaluated a provenance-based APT detector in which every cross-client quantity, not only the model update, is computed under a secure-aggregation protocol, and in which each alert can be accompanied by an edge-level Shapley explanation whose contributions sum exactly to the node's anomaly score. On two hosts of the DARPA TC Engagement 3 benchmark, with a federation simulated from the benign day-graphs of one host, the securely aggregated model ranked the labelled malicious entities above the benign ones with AUROC 0.995 (THEIA) and 0.971 (CADETS), was not measurably different from plain federated averaging, and was at least as good as the centralized reference. Broken down by attack behaviour, the network-scanning fan-out of the implant (T1046) was found completely at the 95% threshold on both hosts, whereas the rare exploitation, loader and command-and-control entities were ranked high (AUROC 0.989 and 0.992) but only partly alarmed at 99%. The secure layer was exact to within 3.2e-07 in the benchmark and 1.19e-07 relative in training, tolerated dropout down to the threshold, cost 15.3 MB of uplink per client for the whole pipeline, and reduced the success of a linkage attack from 85.6% (THEIA) and 99.0% (CADETS) on plain updates to the level of chance on masked uploads.

The study also produced negative and cautionary results that we consider as important as the positive ones. The raw-feature baseline is unreliable; most of the benefit comes from the graph encoder, while edge-type embedding’s and feature masking

show no consistent benefit. A benign-calibrated 99% threshold gives a usable alarm policy on THEIA (F1 0.890) but is brittle on CADETS, where the attack forms a single score cluster at the margin of the threshold. The Shapley explainer is exactly complete and stable, but its fidelity is matched by cheaper methods and its alignment with entity-level labels beyond the incident edges is weak. Attention weights are not an explanation of the detector.

These conclusions are bounded by the evaluation: a 2018 benchmark with scripted attacks and entity-level labels, two hosts, three seeds for the main configurations, a simulated rather than a cross-organization federation, and a semi-honest server. Natural next steps are a heterogeneous multi-host federation, hardening the protocol against a malicious server and adding differential privacy, semantic node features, explanations over paths rather than edges, and evaluation on newer and larger corpora and under adaptive attackers.

Declarations

Acknowledgements
NA

Funding
NA

Contributions
A.F.A; A.S.B.A; Conceptualization, A.F.A; A.S.B.A; Investigation, A.F.A; A.S.B.A; Writing (Original Draft), A.F.A; A.S.B.A; Writing (Review and Editing) Supervision, A.F.A; A.S.B.A; Project Administration.

Ethics declarations
This article does not contain any studies with human participants or animals performed by any of the authors.

Data Availability Statement
NA

Use of Generative Artificial Intelligence
All authors declare that no generative artificial intelligence tools were used in the preparation of this manuscript.

Competing interests
All authors declare no competing interests.

References

[1] Fotouhi, F., Balu, A., Jiang, Z., Esfandiari, Y., Jahani, S., & Sarkar, S. (2024). Dominating Set Model Aggregation for communication-efficient decentralized deep learning. Neural Networks, 171, 25–39. https://doi.org/10.1016/j.neunet.2023.11.057.

[2] Zhang, Y., Li, X., Luo, Q., Wang, Y., & Shen, Y. (2023). Practical and Efficient Secure Aggregation for Privacy-Preserving Machine Learning. Proceedings of the 2023 Asia Conference on Artificial Intelligence, Machine Learning and Robotics, 1–8. https://doi.org/10.1145/3625343.3625344

[3] Kairouz, P., & McMahan, H. B. (2021). Advances and Open Problems in Federated Learning. Foundations and Trends® in Machine Learning, 14(1-2), 1–210. https://doi.org/10.1561/2200000083

[4] Kannan, Y. (2024). Federated Learning in Cybersecurity: Applications, Challenges, and Future Directions. International Journal of Science and Research (IJSR), 13(7), 617–622. https://doi.org/10.21275/mr24706174710

[5] Romhányi, Z. A., & Kummer, A. (2026). Bayesian models in Federated Learning architectures. Machine Learning with Applications, 25, 100949. https://doi.org/10.1016/j.mlwa.2026.100949

[6] Zhao, S., & Keshtkar, S. (2026). Growing Domain-Specific LLMs Through Federated Split-Phase Learning: System Design and Analysis. 2026 2nd International Conference on Federated Learning and Intelligent Computing Systems (FLICS), 346–355. https://doi.org/10.1109/flics70075.2026.11621959

[7] So, J., Guler, B., & Avestimehr, A. S. (2021). Turbo-Aggregate: Breaking the Quadratic Aggregation Barrier in Secure Federated Learning. IEEE Journal on Selected Areas in Information Theory, 2(1), 479–489. https://doi.org/10.1109/jsait.2021.3054610

[8] Girgis, A. M., Data, D., & Diggavi, S. (2021). Differentially Private Federated Learning with Shuffling and Client Self-Sampling. 2021 IEEE International Symposium on Information Theory (ISIT), 338–343. https://doi.org/10.1109/isit45174.2021.9517906

[9] Dwork, C., & Roth, A. (2014). The Algorithmic Foundations of Differential Privacy. Foundations and Trends® in Theoretical Computer Science, 9(3-4), 211–487. https://doi.org/10.1561/0400000042.

[10] Moriai, S. (2019). Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. 2019 IEEE 26th Symposium on Computer Arithmetic (ARITH), 198. https://doi.org/10.1109/arith.2019.00047

[11] Bidargaddi, A. P., D, D. A., & Prasanna, S. R. M. (2025). Enhancing Speaker Representations via Graph Attention Networks and Contrastive Learning. 2025 IEEE Conference on Engineering Informatics (ICEI), 1–8. https://doi.org/10.1109/icei68356.2025.11603690

[12] Wang, J., & Deng, Z. (2021). A Deep Graph Wavelet Convolutional Neural Network for Semi-supervised Node Classification. 2021 International Joint Conference on Neural Networks (IJCNN), 1–8. https://doi.org/10.1109/ijcnn52387.2021.9533634

[13] Huang, Q., Ren, H., Chen, P., Kržmanc, G., Zeng, D., Liang, P., & Leskovec, J. (2023). PRODIGY: Enabling In-context Learning Over Graphs. Advances in Neural Information Processing Systems 36, 16302–16317. https://doi.org/10.52202/075280-0718

[14] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A Comprehensive Survey on Graph Neural Networks. IEEE Transactions on Neural Networks and Learning Systems, 32(1), 4–24. https://doi.org/10.1109/tnnls.2020.2978386

[15] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. AI Open, 1, 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001

[16] Kairouz, P., & McMahan, H. B. (2021). Advances and Open Problems in Federated Learning. Foundations and Trends® in Machine Learning, 14(1-2), 1–210. https://doi.org/10.1561/2200000083

[17] Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph Convolutional Neural Networks for Web-Scale Recommender Systems. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &Amp; Data Mining, 974–983. https://doi.org/10.1145/3219819.3219890

[18] Huang, S., Poursafaei, F., Danovitch, J., Fey, M., Hu, W., Rossi, E., Leskovec, J., Bronstein, M., Rabusseau, G., & Rabbany, R. (2023). Temporal Graph Benchmark for Machine Learning on Temporal Graphs. Advances in Neural Information Processing Systems 36, 2056–2073. https://doi.org/10.52202/075280-0099

[19] Al-shareeda, M., Musa, H. A., Jaafar, A., Adel Salman, A., Tami, Z. J., Hameed, H. M., Muaid, A., Hasan, H. A. H., & Bashkh, N. S. (2026). Design and Implementation of a Speech-to-Sign Robotic Arm for Deaf Communication. International Journal of Cybersecurity Engineering and Innovation, 2026(1), 25. https://doi.org/10.68281/ijcei.theitap.2026.1.3

[20] Alqutaish, A., & Alhumaid, K. (2026). A Federated Learning Framework for Privacy-Preserving Cyber Threat Detection. Journal of Information and Cybersecurity, 2026(1), 32. https://doi.org/10.68281/jics.theitap.2026.1.3

[21] Günnemann, S. (2022). Graph Neural Networks: Adversarial Robustness. Graph Neural Networks: Foundations, Frontiers, and Applications, 149–176. https://doi.org/10.1007/978-981-16-6054-2_8

[22] Schmalhofer, F. (2019). Unified Model of Knowledge Acquisition and Experimental Predictions. Constructive Knowledge Acquisition, 107–130. https://doi.org/10.4324/9781315789439-7

[23] Numeroso, D., & Bacciu, D. (2021). MEG: Generating Molecular Counterfactual Explanations for Deep Graph Networks. 2021 International Joint Conference on Neural Networks (IJCNN), 1–8. https://doi.org/10.1109/ijcnn52387.2021.9534266

[24] Li, J., Zhang, C., & Zhang, C. (2023). Heterogeneous Temporal Graph Neural Network Explainer. Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 1298–1307. https://doi.org/10.1145/3583780.3614909

[25] Liu, L., Wan, P., Zhang, F., Zhang, Y., Liu, Q., & Wang, G. (2025). Learning model-level explanations of graph neural networks via subgraph order embedding space. Neural Networks, 191, 107815. https://doi.org/10.1016/j.neunet.2025.107815

[26] Duval, A., & Malliaros, F. D. (2021). GraphSVX: Shapley Value Explanations for Graph Neural Networks. Machine Learning and Knowledge Discovery in Databases. Research Track, 302–318. https://doi.org/10.1007/978-3-030-86520-7_19

[27] Mastropietro, A., Pasculli, G., Feldmann, C., Rodríguez-Pérez, R., & Bajorath, J. (2022). EdgeSHAPer: Bond-centric Shapley value-based explanation method for graph neural networks. iScience, 25(10), 105043. https://doi.org/10.1016/j.isci.2022.105043

[28] Pasquini, D., Francati, D., & Ateniese, G. (2022). Eluding Secure Aggregation in Federated Learning via Model Inconsistency. Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2429–2443. https://doi.org/10.1145/3548606.3560557

[29] Garg, S., & Torra, V. (2026). PrunePrivyTune: Accelerating Language Models with Pruning and Differentially Private Fine-Tuning. Machine Learning, 115(5). https://doi.org/10.1007/s10994-026-07016-y[29] A. Goyal, X. Han, G. Wang, and A. Bates, "Sometimes, you aren't what you do: Mimicry attacks against provenance graph host intrusion detection systems," in Proc. NDSS, 2023.

[30] Wu, J., Morikawa, T., & Mori, T. (2026). Evading Attention-based Dynamic Malware Detectors: An RL-Guided Method for Adversarial API Parameter Injection. Proceedings of the 41st ACM/SIGAPP Symposium on Applied Computing, 1810–1819. https://doi.org/10.1145/3748522.3779810

[31] Zipperle, M., Gottwalt, F., Chang, E., & Dillon, T. (2022). Provenance-based Intrusion Detection Systems: A Survey. ACM Computing Surveys, 55(7), 1–36. https://doi.org/10.1145/3539605

[32] Inam, M. A., Chen, Y., Goyal, A., Liu, J., Mink, J., Michael, N., Gaur, S., Bates, A., & Hassan, W. U. (2023). SoK: History is a Vast Early Warning System: Auditing the Provenance of System Intrusions. 2023 IEEE Symposium on Security and Privacy (SP), 2620–2638. https://doi.org/10.1109/sp46215.2023.10179405

[33] (Ed.). (2019). Hybrid Graph Convolutional Networks for Semi-Supervised Classification. Proceedings of 2019 the 9th International Workshop on Computer Science and Engineering. https://doi.org/10.18178/wcse.2019.06.016

[34] Zhou, W.-G., & Abbas, K. (2025). Unsupervised Inductive Node Representation Learning for Dynamic Graphs. IEEE Access, 13, 55034–55048. https://doi.org/10.1109/access.2025.3553377

[35] Dhami, D. S., Mathur, S., Yan, S., & Natarajan, S. (2025). Representing Multi-Relational Data via Statistical Relational Learning and Graph Convolutional Networks. Proceedings of the 13th ACM IKDD International Conference on Data Science, 153–163. https://doi.org/10.1145/3799830.3799847

[36] Kadhim, A. F., Hamzah, A. E., Al-Shareeda, M. A., Hussein, A. I., & Sapiee, N. M. (2026). Accurate Network Intrusion Detection using a Feedforward Neural Network and Bee Colony Optimization Algorithm. International Journal of Cybersecurity Engineering and Innovation, 2026(1), 49. https://doi.org/10.68281/ijcei.theitap.2026.1.7

[37] Lan, K., Li, G., Huang, W., Li, J., & Yu, Y. (2026). LLM-HGAN: LLM-Enhanced Heterogeneous Graph Attention Networks for Advanced Persistent Threat Detection. IEEE Transactions on Network Science and Engineering, 13, 8892–8910. https://doi.org/10.1109/tnse.2026.3687554

[38] Albahar, M. A. (2026). A Hybrid Self-Supervised Learning Framework for Advanced Persistent Threat Detection. Computers, Materials &Amp; Continua, 0(0), 1–10. https://doi.org/10.32604/cmc.2026.079941

[39] Krishnapriya, S., Singh, S., & Pramodkumar, A. (2026). FedDTE-IDS: A Federated Dual-Model Deep Transfer Ensemble Framework for Privacy-Preserving Advanced Persistent Threat Detection in Heterogeneous Industrial Internet of Things Networks. IEEE Transactions on Industrial Cyber-Physical Systems, 1–9. https://doi.org/10.1109/ticps.2026.3746211

[40] Almaiah, M. A., Khraisat, A., & Mamodiya, U. (2026). A Hybrid Post-Quantum Cryptographic Framework for Secure 5G/6G and Massive IoT Networks. Journal of Information and Cybersecurity, 2026(1), 45. https://doi.org/10.68281/jics.theitap.2026.1.4

[41] Zhang, K., Sun, L., Ding, B., Yiu, S. M., & Yang, C. (2024). Deep Efficient Private Neighbor Generation for Subgraph Federated Learning. Proceedings of the 2024 SIAM International Conference on Data Mining (SDM), 806–814. https://doi.org/10.1137/1.9781611978032.92

[42] Wu, C., Wu, F., Lyu, L., Qi, T., Huang, Y., & Xie, X. (2022). A federated graph neural network framework for privacy-preserving personalization. Nature Communications, 13(1). https://doi.org/10.1038/s41467-022-30714-9

[43] Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., & Sadeghi, A.-R. (2019). DÏoT: A Federated Self-learning Anomaly Detection System for IoT. 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 756–767. https://doi.org/10.1109/icdcs.2019.00080

[44] Liu, Q., & Feng, Y. (2026). Federated learning and edge computing-based collaborative detection system for IoT anomaly behavior. Discover Artificial Intelligence, 6(1). https://doi.org/10.1007/s44163-026-02041-1