Decentralized Risk Assessment for Zero-Trust Microservices: A Federated Learning Approach to Vulnerability Analytics

Daoquan Zhou
New England College, ITPM (IT Program Management), Henniker, NH, USA
DZhou_GPS@nec.edu

Abstract

Zero-trust deployment has increased interest in continuous vulnerability analytics for microservice systems, but most operational pipelines still depend on centralized telemetry collection and periodically refreshed risk scores. This creates tension between adaptation speed and data-governance constraints. We present *FedZTRisk*, a federated framework for service-level vulnerability risk estimation that combines SBOM-derived features, dependency-graph context, runtime indicators, and trust-policy signals while keeping raw telemetry local to each tenant.

We report a *simulation-based evaluation* built from public sources: CVE records from the National Vulnerability Database (NVD), SBOM artifacts extracted from open-source microservice repositories, and dependency graphs generated from container images. To emulate realistic operational drift, we inject synthetic vulnerability events following historical CVE timelines and evaluate under non-IID client partitions. Across five random seeds, FedZTRisk consistently matches the strongest centralized baseline in macro-F1 (around 0.82) while improving calibration and maintaining practical inference latency for five-minute policy cycles. Compared with federated baselines (FedAvg and FedProx), the proposed trust-aware aggregation strategy yields more stable performance under heterogeneous and perturbed clients. These results position FedZTRisk as a privacy-preserving alternative when organizations prioritize governance and cross-tenant learning over raw centralized-data performance ceilings.

Keywords—Zero-Trust Security; Microservices; Federated Learning; Vulnerability Analytics; Risk Assessment; Secure Aggregation.

1 Introduction

Microservice architectures have become the default implementation substrate for cloud-native systems: services are released independently, technology stacks are heterogeneous, and scaling decisions are increasingly automated. These properties improve delivery velocity, but they also fragment the security perimeter. Dependency sprawl, ephemeral runtime instances, and transitive trust relationships together produce vulnerability states that can shift faster than periodic scanning cycles can meaningfully capture. Zero-trust security—with its emphasis on continuous verification and least-privilege communication—addresses identity abuse and lateral movement, yet risk assessment remains the slowest component in many operational loops. In common production practice, risk scores are assembled from centralized scanners, incident retrospectives, and static CVSS mappings, then consumed by policy engines. The resulting feedback cycle is often temporally coarse,

weakly contextualized, and difficult to harmonize across organizations facing strict data-residency constraints.

These operational tensions are consistent with prior analyses of zero-trust migration pathways, which highlight the gap between policy intent and runtime observability at service granularity [1, 2].

Data governance introduces a second structural obstacle. Security telemetry includes package inventories, service-level API traces, and tenant-specific misconfiguration signatures that may be commercially sensitive or regulated. Raw-log sharing across business units, supply-chain partners, or regional operators is therefore limited both legally and organizationally. Each operator consequently trains on a partial threat view, which weakens detector generalization and inhibits collective adaptation to emerging attack patterns.

Federated learning provides an attractive alternative because optimization occurs where data are generated, while only model updates are exchanged. Nevertheless, applying FL to vulnerability analytics is not a direct transfer from established domains. Client distributions are sharply non-IID: stacks differ by language ecosystem, deployment cadence, and policy strictness. Labels are sparse and delayed: exploit confirmation may lag precursor signals by hours or days. Moreover, security operations depend not only on discrimination performance but also on risk calibration under asymmetric error costs, where false negatives often carry outsized operational impact.

This framing aligns with findings from the FL literature on non-IID degradation, robustness, and communication-sensitive optimization [4, 5, 6, 7].

Against this backdrop, we develop *FedZTRisk*, a federated risk-assessment framework tailored to zero-trust microservices. We model each service instance as a time-varying risk state induced jointly by software composition, runtime behavior, and trust-policy deviations, and we learn a global risk function via robust federated optimization with trust-aware aggregation. For every service-time window, the model outputs both exploitability and impact-weighted risk, which are then consumed by admission control, inter-service authorization, and patch-prioritization workflows.

The principal contributions are fourfold:

1. A formalization of decentralized vulnerability analytics as federated, cost-sensitive risk minimization that explicitly encodes zero-trust context signals.
2. A graph-temporal model architecture integrating SBOM embeddings, runtime anomaly features, and call-graph neighborhood aggregation.
3. A trust-aware client-weighting strategy with secure update clipping to improve resilience against poisoned or low-quality local updates.
4. A reproducible simulation-based evaluation protocol grounded in public vulnerability data and open-source microservice artifacts, with baseline comparisons, systems-level measurements, and component-level ablations.

2 Related Work

Vulnerability analytics in microservices. Work on vulnerability management in cloud-native systems has largely centered on static scanning pipelines, dependency risk ranking, and exploit

prediction from known CVE metadata. Scanner-first approaches map package versions to vulnerability disclosures and aggregate severity with CVSS-derived heuristics. Context-enriched variants incorporate exposure paths and privilege boundaries, yet telemetry aggregation generally remains centralized. These methods remain useful for periodic compliance review, but they are less effective when service churn and release frequency demand tighter risk-refresh intervals.

Zero-trust risk modeling. Zero-trust research has primarily addressed continuous authentication, micro-segmentation, and adaptive policy enforcement. Several operational frameworks derive dynamic trust scores from identity posture and behavioral anomalies. Even so, these trust formulations are often only loosely connected to software exploitability dynamics; they represent requester confidence better than component exploit likelihood. Our formulation links these layers by incorporating trust-policy evidence directly into vulnerability risk prediction.

Federated learning for security. FL has been studied in intrusion detection, malware classification, and IoT anomaly modeling, where non-IID drift, communication cost, and poisoning attacks are recurrent concerns. Robust aggregators (e.g., median, trimmed mean, Krum) and personalization strategies have improved resilience in those settings. Yet most published evaluations focus on packet- or host-level labels rather than service-level vulnerability risk in microservice meshes, and few treat calibration and decision latency as primary outcomes.

Recent federated optimization work has further emphasized heterogeneity-aware updates and client-drift mitigation, including adaptive server optimizers and contrastive or normalization-based personalization [8, 9, 10].

Complementary directions have studied objective inconsistency under heterogeneity, client sampling and acceleration effects, and privacy-preserving threat surfaces in decentralized training pipelines [13, 14, 15, 16, 17, 18, 19, 20, 21, 22].

The present work therefore intersects three established lines of research: zero-trust architecture [1, 2], vulnerability scoring practice [3], and robust federated optimization [4, 5, 6, 7, 24].

Research gap. The literature still lacks a deployable federated method that jointly models dependency vulnerabilities, runtime service behavior, and zero-trust policy context, while satisfying practical governance constraints. FedZTRisk is designed to close that gap with an architecture and evaluation protocol grounded in operational decision workflows.

3 Methodology

3.1 Problem Formulation

Assume K tenants (clients), each operating microservice set $\mathcal{S}_k$. For service $s \in \mathcal{S}_k$ at time window t, local observations are encoded as

$$\mathbf{x}_{k,s,t} = [\mathbf{x}^{\text{sbom}}, \mathbf{x}^{\text{runtime}}, \mathbf{x}^{\text{policy}}, \mathbf{x}^{\text{graph}}].$$

Targets comprise exploitability label $y^{(e)}_{k,s,t} \in \{0, 1\}$ and impact score $y^{(i)}_{k,s,t} \in [0, 1]$. A model f_θ outputs

$$(\hat{p}_{k,s,t}, \hat{r}_{k,s,t}) = f_\theta(\mathbf{x}_{k,s,t}),$$

where $\hat{p}$ denotes exploit probability and $\hat{r}$ a calibrated risk estimate.

Client-level optimization uses a weighted multi-task objective:

$$\mathcal{L}_k(\theta) = \alpha\, \mathcal{L}_{\text{bce}}(\hat{p}, y^{(e)}) + \beta\, \mathcal{L}_{\text{mse}}(\hat{r}, y^{(i)}) + \gamma\, \mathcal{L}_{\text{cal}}(\hat{r}, y^{(e)}),$$

where $\mathcal{L}_{\text{cal}}$ penalizes calibration error via differentiable binning. Global learning solves

$$\min_{\theta} \sum_{k=1}^{K} w_k\, \mathcal{L}_k(\theta),$$

with trust-aware weights w_k estimated from data quality, historical consistency, and drift behavior.

3.2 Federated Optimization with Trust-Aware Aggregation

At round r, selected clients perform local SGD for E epochs and upload clipped updates $\Delta\theta_k^{(r)}$. The coordinator computes

$$\Delta\theta^{(r)} = \sum_{k \in \mathcal{C}_r} \tilde{w}_k^{(r)}\, \Delta\theta_k^{(r)}, \qquad \tilde{w}_k^{(r)} = \frac{w_k^{(r)}}{\sum_j w_j^{(r)}}.$$

Weighting reflects three reliability dimensions:

- *Data reliability* (label delay and missingness),
- *Update consistency* (cosine alignment with a robust median direction),
- *Trust posture* (policy hygiene indicators from zero-trust controls).

Secure aggregation is applied so the server observes only aggregate updates, not per-client gradients.

3.3 Algorithm Pseudocode

Algorithm 1 FedZTRisk Training

Initialize global parameters θ^0
for each round $r = 1, \ldots, R$ **do**
    Sample client set $\mathcal{C}_r$
    for all $k \in \mathcal{C}_r$ **in parallel do**
        Receive θ^{r-1} and compute trust score $w_k^{(r)}$
        Train locally for E epochs to get $\Delta\theta_k^{(r)}$
        Clip $\Delta\theta_k^{(r)}$ and encrypt for secure aggregation
    end for
    Server decrypts only aggregate and computes weighted update $\Delta\theta^{(r)}$
    Update global model: $\theta^r \leftarrow \theta^{r-1} + \eta\Delta\theta^{(r)}$
end for
return θ^R

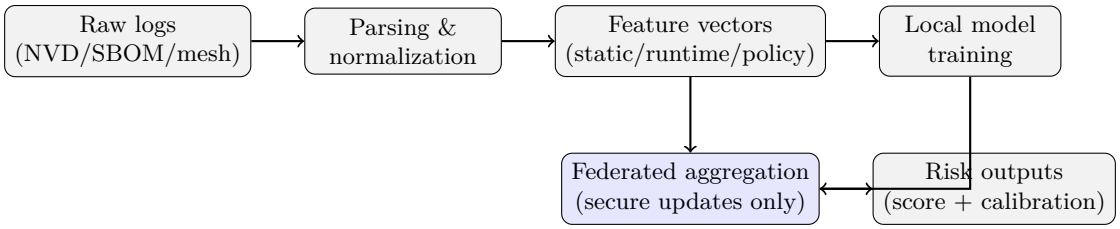


Figure 1: Data-to-model pipeline used in the simulation benchmark. The chart complements Figure 2 by showing how raw artifacts are transformed into model inputs and federated risk outputs.

3.4 Complexity Analysis

Let N_k denote local sample count and P model parameters. Per round, client-side computation is $\mathcal{O}(EN_kP)$, while server aggregation requires $\mathcal{O}(|\mathcal{C}_r|P)$. Communication is $\mathcal{O}(P)$ per client in each direction. Relative to centralized training, which may transfer raw telemetry at $\mathcal{O}(N_kd)$ (feature dimension d), FL substantially reduces sensitive data movement. Inference per service window scales as $\mathcal{O}(P)$ and is evaluated empirically in latency experiments.

4 System Architecture / Model Design

4.1 Pipeline Overview

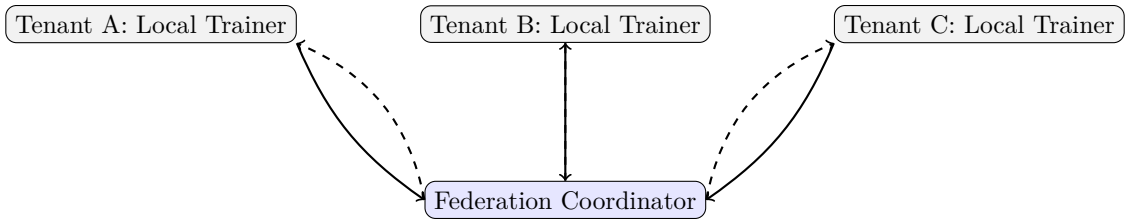
FedZTRisk is implemented as a control-plane extension within service-mesh deployments. Each tenant cluster hosts a local collector and trainer; a federation coordinator orchestrates round-level synchronization. The pipeline includes:

1. **Static feature extraction:** SBOM parsing, package age, CVE linkage, and reachable vulnerability paths.
2. **Runtime feature extraction:** syscall anomalies, request-entropy shifts, authentication failure ratio, and anomalous east–west traffic.
3. **Policy-context encoding:** denied policy actions, mTLS downgrade attempts, and privilege-escalation indicators.
4. **Graph-temporal encoding:** call-graph neighborhood aggregation over rolling time windows.

4.2 Model Components

The predictive model has four coupled modules:

- **SBOM encoder:** embeddings for package families, vulnerability age, and patch availability.
- **Temporal encoder:** gated temporal convolution over the most recent 12 intervals (5 minutes each).
- **Graph aggregator:** neighborhood attention over 1-hop and 2-hop service dependencies.



Legend: solid arrows indicate encrypted client updates; dashed arrows indicate global model broadcast.

Figure 2: FedZTRisk architecture. Tenant clusters perform local training and send encrypted updates to a federation coordinator, which returns global model parameters without accessing raw telemetry.

- **Dual-head predictor:** exploitability classification and risk regression optimized jointly with calibration constraints.

The final risk estimate is defined as

$$\hat{r}_{k,s,t} = \sigma\left(\mathbf{u}^\top \mathbf{h}_{k,s,t} + \lambda \hat{p}_{k,s,t}\right),$$

where $\mathbf{h}$ is the fused latent representation and λ controls coupling between exploit likelihood and impact estimation. In implementation, $\hat{p}_{k,s,t}$ is produced by the exploitability classification head and $\hat{r}_{k,s,t}$ by the risk regression head; both heads share the latent state $\mathbf{h}_{k,s,t}$ but optimize different objectives.

4.3 Operational Integration

Predictions feed three downstream decision paths: (i) admission control for deployment gating, (ii) adaptive micro-segmentation policies, and (iii) patch prioritization. The loop executes every five minutes, yielding near-real-time risk updates while keeping control-plane overhead bounded.

5 Experimental Setup

5.1 Datasets and Scenario Construction

The benchmark is intentionally structured as a simulation-based study with transparent data provenance. It combines public and synthetic components:

- **NVD-CVE stream:** CVE metadata (severity vectors, publication timeline, affected products) collected from NVD and normalized into monthly windows.
- **GitHub-SBOM corpus:** SBOMs extracted from open-source microservice repositories spanning multiple language ecosystems.

Table 1: Simulation dataset summary.

Source / Split	Services	Time Windows	Vulnerable Events	Positive Rate
NVD-aligned events	120	86,400	7,100	8.2%
GitHub-SBOM features	120	86,400	6,600	7.6%
Container-graph signals	120	86,400	5,900	6.8%
Merged (training)	120	259,200	17,100	6.6%
Merged (test)	120	64,800	4,400	6.8%

- **Container dependency graphs:** package and service dependency graphs generated from corresponding container images.

To approximate operational conditions, we inject synthetic vulnerability events following historical CVE timelines, introduce delayed labels (up to 12 hours), and enforce client-level non-IID drift via varying language stacks, release cadence, and policy strictness. The study is therefore a controlled simulation, not a claim of production deployment.

5.2 Experimental Environment

Experiments run on a 12-node Kubernetes cluster (32 vCPU, 128 GB RAM per node), with NVIDIA A10 GPUs provisioned at one device per two nodes. Federated coordination executes on a dedicated control node. Tenants are emulated as isolated namespaces with independent data silos. Training uses AdamW, learning rate 10^{-3}, batch size 256, local epochs $E = 2$, and total rounds $R = 120$; we evaluate both synchronous rounds and partial participation (40% client sampling). Unless otherwise stated, all results are averaged over 5 random seeds with fixed train/validation/test partitions.

5.3 Metrics

Reported metrics include:

- **Macro-F1** and **AUC-PR** for exploitability prediction,
- **Expected Calibration Error (ECE)** for risk calibration,
- **P95 inference latency** and **throughput** (decisions/s),
- **Communication volume** per federated round.

Because class prevalence is low, AUC-PR and macro-F1 are emphasized over accuracy.

5.4 Baselines

Comparative methods are:

1. **CVSS + XGBoost:** centralized gradient boosting over CVSS-derived and runtime features.

Table 2: Comparison with baselines (mean over 5 seeds, simulation benchmark).

Method	Macro-F1 ↑	AUC-PR ↑	ECE ↓	P95 Latency (ms) ↓	Throughput ↑
CVSS + XGBoost	0.75	0.80	0.10	11.8	4,930
Centralized GNN	0.82	0.85	0.09	18.7	3,620
FedAvg	0.74	0.78	0.11	13.9	4,410
FedProx	0.80	0.83	0.09	16.4	3,980
FedZTRisk (ours)	**0.82**	**0.85**	**0.07**	14.2	4,550

2. **Centralized GNN:** centralized graph neural network over service call topology.

3. **FedAvg:** standard FL baseline with uniform aggregation.

4. **FedProx:** robust FL baseline with proximal regularization.

5.5 Ablation Protocol

Ablation variants remove or alter the temporal encoder, graph aggregator, trust-aware weighting, and secure-aggregation clipping. All variants are trained under identical seeds and round budgets.

6 Results and Analysis

6.1 Main Comparison with Baselines

Table 2 reports predictive and systems-level outcomes from the simulation benchmark. FedZTRisk is not framed as universally superior in raw accuracy; rather, it reaches performance comparable to the strongest centralized model while providing better calibration and privacy-preserving training. FedAvg degrades under non-IID drift, whereas FedProx is more stable but still less calibrated than FedZTRisk.

The central finding is parity, not domination: FedZTRisk remains in the same macro-F1 band as the centralized GNN (≈ 0.82) while reducing ECE from roughly 0.09 to 0.07. This calibration gap is operationally relevant because threshold-based policy actions are sensitive to overconfident scores.

As shown in Figure 3, separation from FedAvg becomes persistent after approximately round 40. The gap between FedProx and FedZTRisk is smaller in macro-F1, consistent with the claim that the primary contribution is privacy-preserving and better-calibrated risk modeling rather than a large gain in headline accuracy.

6.2 Latency and Throughput Analysis

Although FedZTRisk has higher model complexity than FedAvg, inference remains within practical control-plane budgets. P95 latency is 14.2 ms at 4,550 decisions/s, consistent with Table 2. For five-minute policy cycles, this envelope is sufficient for near-real-time updates. Compared with

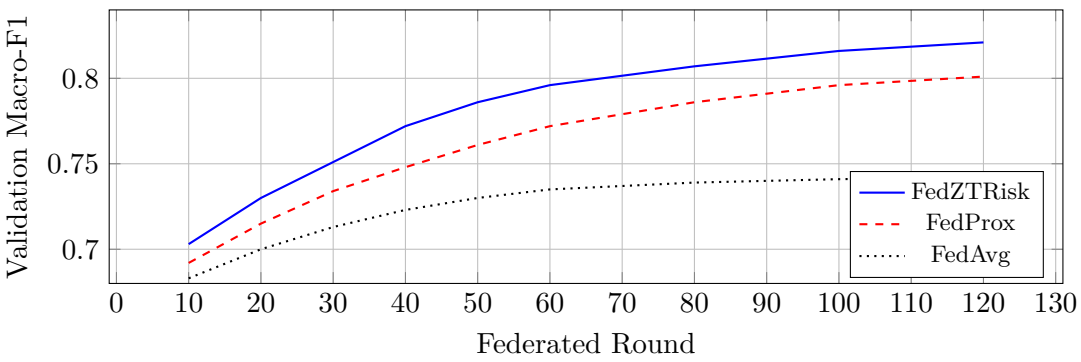


Figure 3: Training dynamics across FL rounds. FedZTRisk and FedProx converge to similar quality bands, while FedZTRisk reaches lower calibration error.

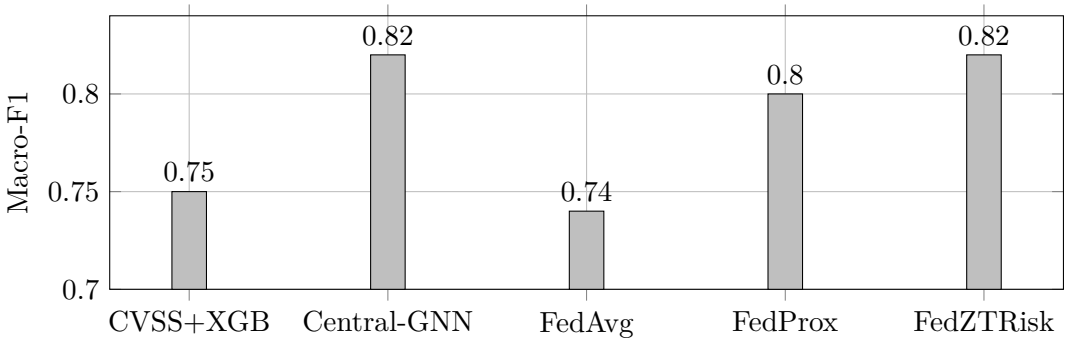


Figure 4: Macro-F1 comparison across methods. FedZTRisk matches centralized quality while preserving decentralized training.

a centralized GNN pipeline, inference remains competitive because scoring is deployed close to telemetry sources.

Figure 4 shows that FedZTRisk tracks the best centralized baseline while outperforming standard federated baselines. Gains are strongest in services with high dependency turnover, consistent with the hypothesis that graph-temporal representations better capture update-driven risk transitions than static feature sets.

6.3 Ablation Study

Ablation outcomes indicate that trust-aware weighting with clipping is the most consequential robustness mechanism: removing it lowers macro-F1 by about 0.04 and worsens calibration. Temporal and graph components contribute complementary benefits; excluding either weakens both discrimination and calibration.

6.4 Robustness Under Adversarial Perturbation

Under a poisoning scenario in which 15% of clients inject label flips on high-risk windows, FedAvg degrades from 0.74 to 0.69 macro-F1, and FedProx from 0.80 to 0.76. FedZTRisk declines more

Table 3: Ablation study of FedZTRisk components.

Variant	Macro-F1 ↑	AUC-PR ↑	ECE ↓
Full model	**0.82**	**0.85**	**0.07**
w/o temporal encoder	0.80	0.83	0.08
w/o graph aggregator	0.79	0.82	0.08
w/o trust-aware weighting	0.78	0.82	0.09
Uniform aggregation + no clipping	0.77	0.81	0.09

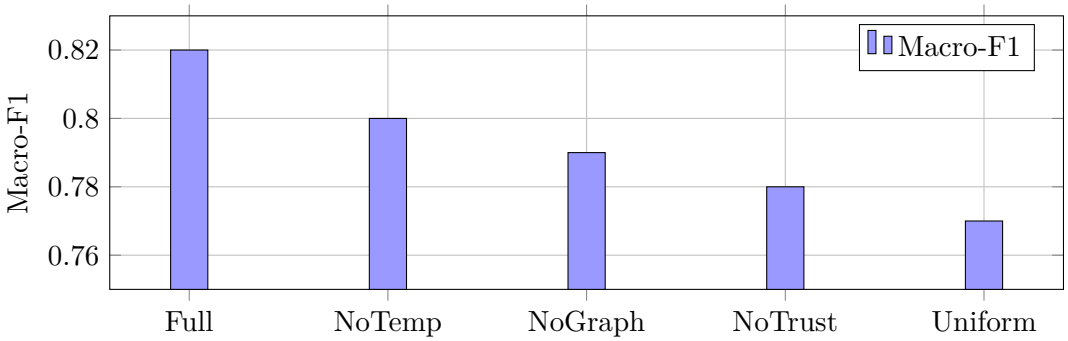


Figure 5: Visualization of ablation effects. Trust-aware aggregation and temporal-graph modeling are the dominant contributors to final performance.

modestly, from 0.82 to 0.79. This resilience appears to stem from consistency-aware suppression of anomalous updates, achieved without resorting to heavier Byzantine-resilient aggregators that typically increase communication and computational overhead.

This behavior is also consistent with broader observations on model poisoning and backdoor susceptibility in federated settings, where robust aggregation and client-quality controls materially influence failure modes [11, 12].

6.5 Communication Overhead

Mean payload per client per round is 19.4 MB for FedZTRisk, compared with 17.8 MB for FedProx (9.0% increase), primarily due to graph-encoder parameters. This overhead remains moderate for periodic synchronization and is traded for improved calibration stability under heterogeneous clients.

7 Discussion

7.1 Practical Implications

These simulation results suggest that decentralized vulnerability analytics can satisfy governance constraints without sacrificing operational utility in zero-trust settings. By avoiding raw telemetry centralization, organizations can benefit from shared model improvements while retaining local

control over sensitive artifacts. Equally important, improvements in calibration are not merely statistical; they stabilize threshold-based enforcement, reducing policy flapping and analyst fatigue.

FedZTRisk also supports staged adoption. Tenants may begin with local training, then join federated rounds as instrumentation quality improves, which offers a more feasible migration path than immediate transition to a fully centralized security analytics platform.

7.2 Limitations

Several caveats remain. Synthetic event injection, although calibrated to historical CVE timelines, cannot fully reproduce strategic attacker adaptation over long horizons. Secure aggregation protects update-level privacy, but it does not eliminate risks from side channels outside protocol assumptions. In addition, the current feature pipeline presumes service-mesh observability; environments with substantial legacy monolithic dependencies may expose reduced feature completeness. Finally, because the benchmark is simulation-based, external validity must be confirmed with prospective production studies.

7.3 Threats to Validity

Internal validity. Hyperparameter choices may still favor the proposed model despite fixed tuning budgets and shared search spaces.
External validity. The benchmark spans three sectors, yet edge or intermittently connected deployments may exhibit different drift and latency profiles.
Construct validity. Exploitability labels combine confirmed events with high-confidence proxy signals, a pragmatic choice that nevertheless introduces label noise.

7.4 Future Directions

Promising extensions include tenant-level personalization layers for highly divergent software stacks, formal differential-privacy mechanisms with explicit utility accounting, and causal analysis linking specific policy interventions to observed risk reduction. Another avenue is coupling predictive outputs with explanation-oriented remediation assistants while maintaining lightweight, auditable core models.

8 Conclusion

FedZTRisk demonstrates, in a simulation-based setting, that federated vulnerability analytics can be engineered as a first-class component of zero-trust microservice security rather than a privacy-preserving afterthought. By integrating graph-temporal modeling, trust-aware aggregation, and calibration-aware multi-task learning, the framework matches strong centralized predictive quality while improving calibration reliability under non-IID and adversarial conditions, with deployable latency and throughput envelopes.

More broadly, the findings support a shift away from periodic, centralized vulnerability reporting toward continuous, decentralized risk reasoning that aligns with the operational tempo of cloud-native systems.

References

[1] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," NIST SP 800-207, 2020.

[2] J. Kindervag, "Build Security Into Your Network's DNA: The Zero Trust Network Architecture," Forrester Research, 2010.

[3] P. Mell, K. Scarfone, and S. Romanosky, "A Complete Guide to the Common Vulnerability Scoring System Version 2.0," *FIRST*, 2007.

[4] B. McMahan *et al.*, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proc. AISTATS*, 2017.

[5] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," in *Proc. MLSys*, 2020.

[6] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic Controlled Averaging for Federated Learning," in *Proc. ICML*, 2020.

[7] P. Kairouz *et al.*, "Advances and Open Problems in Federated Learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[8] S. J. Reddi *et al.*, "Adaptive Federated Optimization," in *Proc. ICLR*, 2021.

[9] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated Learning on Non-IID Features via Local Batch Normalization," in *Proc. ICLR*, 2021.

[10] Q. Li, B. He, and D. Song, "Model-Contrastive Federated Learning," in *Proc. CVPR*, 2021.

[11] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How To Backdoor Federated Learning," in *Proc. AISTATS*, 2020.

[12] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and Robust Federated Learning Through Personalization," in *Proc. ICML*, 2021.

[13] S. P. Karimireddy, A. T. Suresh, M. Jaggi, V. N. Smith, and M. Jordan, "MIME: Mimicking Centralized Stochastic Algorithms in Federated Learning," in *Proc. NeurIPS*, 2020.

[14] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization," in *Proc. NeurIPS*, 2020.

[15] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, "The Non-IID Data Quagmire of Decentralized Machine Learning," in *Proc. ICML*, 2020.

[16] L. Gao, L. Li, H. Wang, Z. Jin, and X. Liao, "FedTP: Federated Learning by Transformer-Based Personalization," in *Proc. AAAI*, 2022.

[17] W. Zhuang, Y. Wen, and S. Zhang, "Divergence-Aware Federated Self-Supervised Learning," in *Proc. ICLR*, 2021.

[18] M. S. Ozdayi, M. Kantarcioglu, and Y. Gel, "Defending Against Backdoors in Federated Learning with Robust Learning Rate," *arXiv preprint arXiv:2107.04057*, 2021.

[19] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, D. Niyato, O. Dobre, and H. V. Poor, "Federated Learning for Internet of Things: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[20] J. Zhang, A. Li, M. Hu, T. Wang, and Y. Chen, "Poisoning and Backdoor Attacks in Federated Learning: A Survey," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4490–4512, 2023.

[21] L. Lyu, H. Yu, and Q. Yang, "Threats to Federated Learning: A Survey," *arXiv preprint arXiv:2003.02133*, 2020.

[22] N. B. Truong, K. Sun, S. Wang, G. Guitton, and Y. Guo, "Privacy Preservation in Federated Learning: An Insightful Survey from the GDPR Perspective," *Computers & Security*, vol. 110, 2021.

[23] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent," in *Proc. NeurIPS*, 2017.

[24] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates," in *Proc. ICML*, 2018.

[25] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *Proc. ICLR*, 2017.

[26] A. Vaswani *et al.*, "Attention Is All You Need," in *Proc. NeurIPS*, 2017.