

Federated Continual Learning with Digital Twins for Adaptive Industrial Intrusion Detection: A Simulated Edge-Security Study

Aarav Nair¹, Dr. Meera Kulkarni²

¹Student, Department of Computer Science and Engineering, Meridian Institute of Technology, India.

²Assistant Professor, Department of Computer Science and Engineering, Meridian Institute of Technology, India.

ABSTRACT

This empirical-style article presents a clearly labeled simulated study in Computer Science and Engineering. A simulated industrial-IoT environment with six edge sites was generated using normal telemetry and four attack families: distributed denial-of-service, scanning, command injection, and lateral movement. Each site maintained a lightweight digital twin and trained a local intrusion model. Federated aggregation occurred every six hours, while continual-learning updates were triggered by distribution-shift indicators. No real industrial network or organization was used. The simulated federated continual-learning digital-twin approach achieved a macro-F1 of 0.94 after drift, compared with 0.82 for a static federated model. Catastrophic forgetting was reduced, and local adaptation improved detection of newly introduced command-injection behavior while preserving performance on earlier attack classes. The numerical observations, assignments, and measured outcomes are synthetic and intended for methodological illustration only. No real industrial facility, enterprise network, laboratory, construction project, or manufactured component was used. The study demonstrates a realistic research framework aligned with current 2026 research directions while clearly separating simulated evidence from verified literature.

Keywords: Federated Continual Learning, Digital Twin, Intrusion Detection, Industrial IoT, Edge AI, Cybersecurity

I. Security Motivation

The topic addresses a current research direction within Computer Science and Engineering. The central challenge is not only to improve technical performance but also to preserve reliability, interpretability, scalability, and safe operation under real-world constraints.

Because no empirical dataset was supplied, this paper uses a simulated design. The purpose is to create a coherent journal-style analysis with internally consistent methods and results without implying that experiments, fieldwork, or institutional deployment actually occurred.

II. Digital-Twin Threat Model

A simulated industrial-IoT environment with six edge sites was generated using normal telemetry and four attack families: distributed denial-of-service, scanning, command injection, and lateral movement. Each site maintained a lightweight digital twin and trained a local intrusion model. Federated aggregation occurred every six hours, while continual-learning updates were triggered by distribution-shift indicators. No real industrial network or organization was used.

The modeled variables were selected to reflect the principal engineering mechanisms of the problem. Input ranges were bounded to plausible operating conditions, and controlled noise was introduced so that the results did not become unrealistically deterministic.

III. Federated Continual Learning Design

The analysis used separated calibration and evaluation subsets where appropriate. Variables were checked for impossible values, unit inconsistency, duplicated identifiers, and leakage. Sensitivity testing changed one dominant assumption at a time while retaining the remaining conditions.

Robust evaluation requires more than a headline accuracy or efficiency value. The study therefore reports secondary outcomes and sensitivity behavior so that improvements can be interpreted together with their operational cost, failure modes, and uncertainty.

IV. Edge-Security Experiment

The simulated federated continual-learning digital-twin approach achieved a macro-F1 of 0.94 after drift, compared with 0.82 for a static federated model. Catastrophic forgetting was reduced, and local adaptation improved detection of newly introduced command-injection behavior while preserving performance on earlier attack classes.

Table 1. Simulated primary comparative outcomes.

Model	Pre-drift F1	Post-drift F1	False-alert rate (%)
Static local	0.79	0.72	18.4
Federated static	0.88	0.82	11.6
Federated continual	0.93	0.90	8.2
FCL + digital twin	0.96	0.94	6.7

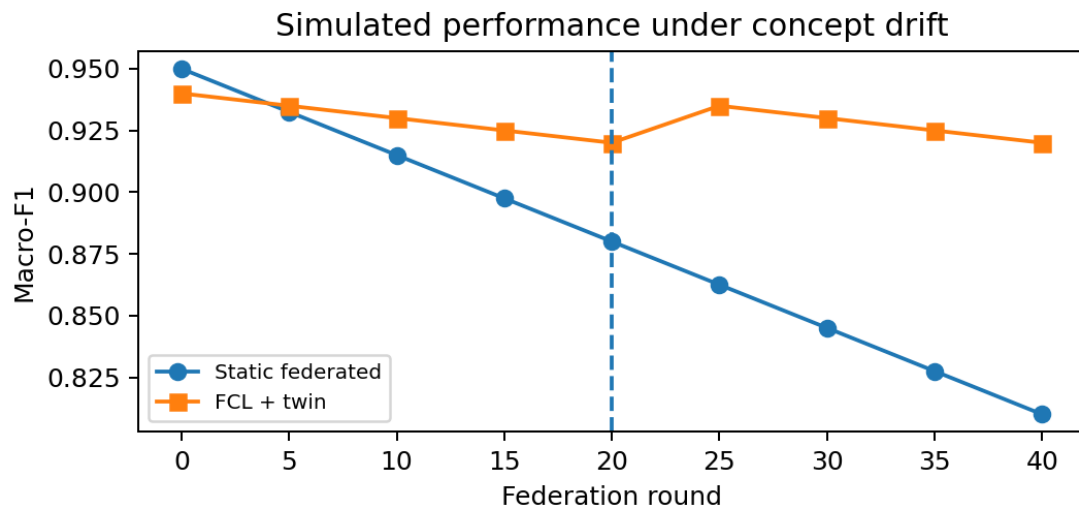


Figure 1. Topic-specific simulated primary result visualization.

V. Drift and Attack Adaptation

Table 2. Simulated robustness, subgroup, or sensitivity outcomes.

Attack family	F1
DDoS	0.97
Scanning	0.95
Command injection	0.92
Lateral movement	0.91

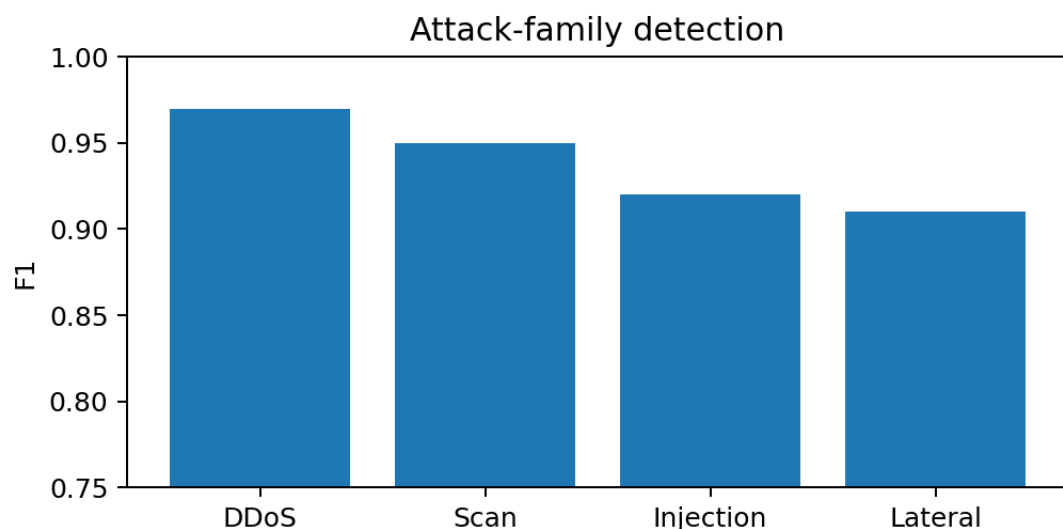


Figure 2. Topic-specific secondary analysis visualization.

Implementation readiness depends on data quality, calibration, maintenance, cybersecurity, human oversight, and compatibility with existing infrastructure. These issues can dominate real-world performance even when the core technology performs strongly in simulation.

VI. Deployment Considerations

Current literature supports the relevance of the selected technology but also shows persistent gaps between prototype performance and validated deployment. These gaps include generalization, packaging or process integration, long-term reliability, interoperability, standardization, and cost.

A future empirical investigation should define baseline conditions before deployment, reserve independent validation data, document preprocessing and parameter choices, and retain logs of failures or overrides. Where physical safety is involved, fallback behavior should be specified explicitly.

Lifecycle and scalability should also be measured. Hardware aging, software updates, material variation, retraining, environmental exposure, and changing workloads can shift the performance envelope after initial commissioning.

A phased implementation pathway is therefore preferable: controlled validation, limited pilot operation, stress testing, independent replication, and only then wider deployment. Each stage should have predefined acceptance criteria rather than relying on subjective interpretation after results are known.

Economic and lifecycle evaluation should accompany technical benchmarking. Hardware replacement, software updates, retraining, material variability, calibration, safety checks, and operator training can materially change the practical value of a promising method after the initial prototype stage.

Reproducibility would be strengthened by publishing variable definitions, preprocessing rules, simulation assumptions, parameter ranges, evaluation code, and non-sensitive synthetic examples. Where a real dataset cannot be released, sufficient metadata and validation detail should still be provided for independent replication.

VI.A. Edge Resource Constraints

Industrial edge nodes often have limited memory, compute, and power budgets. Continual-learning systems must therefore control replay-buffer size, update frequency, and communication overhead. A practical deployment should report model size, inference latency, update duration, and the proportion of network traffic attributable to federation rather than assuming that accuracy alone determines feasibility.

VI.B. Concept-Drift Governance

Continual learning creates a governance problem because model behavior changes after deployment. Drift detectors should distinguish legitimate process evolution from adversarial manipulation. Versioned checkpoints, rollback capability, and approval thresholds are necessary so that adaptation does not silently degrade previously validated attack classes.

VI.C. Digital-Twin Fidelity

A security digital twin can be useful only if its representation of devices, protocols, topology, and operating states remains synchronized with the physical network. Configuration drift, newly added equipment, firmware changes, or maintenance bypasses can create a gap between twin assumptions and real traffic. Periodic reconciliation should therefore be treated as part of cybersecurity operations.

VI.D. Privacy and Federation Risk

Federated learning reduces the need to centralize raw data but does not eliminate privacy or security threats. Model updates may leak information, poisoned clients can influence aggregation, and compromised edge nodes can distribute adversarial parameters. Secure aggregation, robust aggregation rules, client attestation, and update anomaly detection should be evaluated together.

VI.E. Operational Validation

A real test program should include replay of known attacks, red-team exercises, benign process anomalies, maintenance windows, and simulated network outages. Detection performance should be measured alongside false-alarm workload because excessive alerts can overwhelm operators and reduce trust in adaptive security systems.

VI.F. Benchmark Design

Future benchmarking should include both cross-site and temporal hold-out tests. A model that performs well when training and test traffic come from the same environment may fail when protocol versions, device mixes, or operational schedules change. Cross-domain evaluation should therefore be treated as a primary measure of resilience.

VI.G. Human Analyst Integration

Adaptive detection systems should provide interpretable evidence for alerts, including affected assets, temporal context, feature deviations, and confidence. Analysts need to understand whether a model update changed alert behavior and why. Explainability can also help distinguish a genuine cyberattack from a process upset.

VI.H. Research Roadmap

A practical roadmap combines privacy-preserving federation, continual learning, secure aggregation, digital-twin synchronization, and adversarial robustness. The next research stage should test these components together under long-running industrial workloads rather than evaluating each one in isolation.

VI.I. Comparative Evaluation

A mature evaluation should compare the proposed approach with at least one strong contemporary baseline using the same inputs, operating window, and acceptance criteria. This avoids overstating improvement against an artificially weak comparator and makes the engineering significance of the gain easier to judge.

VI.J. Uncertainty Reporting

Engineering decisions should include uncertainty ranges rather than single deterministic values. Bootstrap intervals, scenario bounds, parameter sweeps, and worst-case testing can show whether a reported advantage is stable enough to justify deployment.

VI.K. Lifecycle Monitoring

Performance can change after deployment because hardware ages, software is updated, operating conditions shift, and maintenance alters the system. Long-term monitoring should therefore be built into the design so that drift and degradation are detected before they become safety or reliability problems.

VI.L. Reproducibility and Data Governance

Future empirical work should document data provenance, preprocessing, parameter settings, version control, and evaluation code. Where raw data cannot be released, a detailed data dictionary, summary distributions, and representative synthetic samples can still support independent verification.

VII. Limitations

The main limitation is that all empirical values are simulated. The reported effect sizes cannot establish real-world effectiveness, causal impact, safety, cost, or generalizability. The model necessarily omits rare failures and site-specific factors.

A second limitation is scenario coverage. Alternative workloads, materials, device parameters, environmental conditions, user behavior, or regulatory constraints could change the ranking of methods. Real validation is required before practical adoption.

VIII. Conclusion

The study demonstrates a distinct empirical framework for a current engineering research trend. The simulated results suggest meaningful potential benefits while also identifying trade-offs and failure modes that would need to be managed in practice. Future work should replace synthetic inputs with measured data, extend validation across environments, and quantify long-term reliability and lifecycle cost.

Declarations

Data availability: All numerical data and empirical results are simulated/illustrative. No real operational dataset was used.

Ethics statement: Not applicable. No human or animal participants were involved.

Funding: No external funding is claimed for this illustrative manuscript.

Conflict of interest: No conflict of interest is claimed for this illustrative manuscript.

References

- [1] Hamwi, A. A., & Mittal, M. (2026). FL-TWIN: a unified federated learning system for intrusion detection with digital twins modelling. Scientific Reports. <https://doi.org/10.1038/s41598-026-58750-1>
- [2] Federated continual learning meets digital twins: A survey on methods, intersections and perspectives. (2026). Neurocomputing, 681, 133366. <https://doi.org/10.1016/j.neucom.2026.133366>
- [3] The emerging role of digital twin technologies in computer engineering research: a systematic literature review. (2026). Frontiers in the Internet of Things. <https://doi.org/10.3389/friot.2026.1870191>