Detection of Misreporting Attacks on Software-Defined Immersive Environments

Sourya Saha, Md. Nurul Absur, Shima Yousefi, Saptarshi Debroy
City University of New York
Emails: {*ssaha2,mabsur,syousefi*}*@gradcenter.cuny.edu, saptarshi.debroy@hunter.cuny.edu*

***Abstract*—The ability to centrally control network infrastructure using a programmable middleware has made Software-Defined Networking (SDN) ideal for emerging applications, such as immersive environments. However, such flexibility introduces new vulnerabilities, such as switch misreporting led load imbalance, which in turn make such immersive environment vulnerable to severe quality degradation. In this paper, we present a hybrid machine learning (ML)-based network anomaly detection framework that identifies such stealthy misreporting by capturing temporal inconsistencies in switch-reported loads, and thereby counter potentially catastrophic quality degradation of hosted immersive application. The detection system combines unsupervised anomaly scoring with supervised classification to robustly distinguish malicious behavior. Data collected from a realistic testbed deployment under both benign and adversarial conditions is used to train and evaluate the model. Experimental results show that the framework achieves high recall in detecting misreporting behavior, making it effective for early and reliable detection in SDN environments.**

***Index Terms*—Software-defined networking, load balancing, virtual reality, quality of experience, misreporting attacks.**

I. Introduction

Software-Defined Networking (SDN) has become a key technology for next-generation networks due to its centralized control, real-time programmability, and dynamic flow management. Such flexibility enables intelligent orchestration of resources, making SDN attractive for performance-critical applications. Immersive 3D applications, such as Virtual and Augmented Reality (VR/AR), is a prominent domain which can leverage SDN to enhance situational awareness in areas such as cyber-training, healthcare, and emergency response [1]. For instance, SDN has been applied to improve VR streaming through MCTS-based routing [2], multipath delivery for tiled content [3], and ML-driven network slicing for latency-sensitive VR [4].

However, SDN's centralized design and hardware-software coupling create a broad attack surface. Known attacks include flow table overflow, packet injection, and traffic flooding, which can destabilize both SDN systems and hosted critical applications. A particularly subtle vulnerability is *misreporting*, wherein a malicious switch(s) falsifies load statistics to bias the controller's flow assignments. This leads to unfair traffic distribution and performance degradation in latency-sensitive workloads. In VR, such manipulation can degrade quality of experience (QoE) by redirecting tasks to compromised servers. Such manipulated pose updates may lead to visually inconsistent rendering, emphasizing the need for timely detection.

While defense mechanims, such as rDefender [5], counterflow [6], and SDN-Guard [7] address rule-based vulnerabilities, effective strategies against switch misreporting attacks are fewer to almost none. Detection of such attacks is challenging because misreported statistics mimic historical patterns, evading threshold-based or anomaly filters. Recent work [8] shows how compromised switches can attract traffic without triggering alarms, exposing a gap in current SDN defenses that largely assume a trusted data plane.

In this paper, we study stealthy misreporting in SDN and its impact on latency-sensitive applications. Using VR offloading as a case study, we deploy our setup on the NSF FABRIC testbed [9], replicating the misreporting model of [8] to show how falsified load reports can redirect workflows to a malicious edge server that perturbs VR pose updates. Integrated with the ILLIXR environment [10], our setup traces the attack's effect from network telemetry to application-level QoE. To counter this, we propose a hybrid detection framework combining statistical features with temporal modeling to reveal anomalies that appear plausible individually but suspicious over time. A transformer-based autoencoder trained on benign telemetry yields unsupervised anomaly signals—reconstruction error, Mahalanobis distance, and rolling-window statistics (z-score, skewness, kurtosis). These signals are then fused with lightweight supervised models—an MLP and calibrated LightGBM—balancing generalization with precision in detecting subtle misreporting.

We evaluate the proposed framework on an SDN testbed built on FABRIC with controlled misreporting. ILLIXR environment serves as a VR layer, offloading head pose estimation to edge servers hosted on FABRIC, letting us measure QoE degradation when workflows are steered to a compromised server. The setup demonstrates higher Absolute Trajectory Error (ATE) and Relative Pose Error (RPE), confirming impact on user experience. Benchmarks, such as precision, recall, F1, and ROC AUC, show up to 20% F1 gain over unsupervised baselines with latency suitable for real-time deployment. System robustness is also assessed under varying window sizes, strides, VR durations, and attack persistence, providing early evidence of effective misreporting detection in SDN-hosted immersive environments.

This material is based upon work supported by the National Science Foundation (NSF) under Award Number CNS-2401928.

978-3-903176-75-1 ©2025 IFIP

The remainder of this paper is organized as follows. Section II reviews relevant prior work. Section III presents the system and threat models. Section IV outlines the proposed detection framework. Section V presents our evaluation methodology and experimental results. Section VI concludes the paper and discusses future work.

II. Related Works

Falsification-based threats remain a major challenge in SDN and cyber-physical systems (CPS), where accurate reporting from distributed components is critical. These attacks exploit trust in telemetry—whether switch statistics, sensor readings, or link metrics—to mislead controllers or higher-level logic. In SDN, the consequences are especially severe given centralized control. The Marionette attack [11] shows how high-priority flow entries can redirect LLDP packets and fabricate a fake but plausible topology. Similarly, the Link Latency Attack (LLA) [12] manipulates latency estimates through ARP flooding and LLDP relaying. Such examples highlight the fragility of data-driven decision-making in programmable networks.

Within this broader class of falsification attacks, misreporting attacks at the SDN data plane are particularly insidious: compromised switches falsify traffic statistics to deceive the controller. Our work builds on [8], which introduced a trivial zero-reporting attack and a stealthier variant drawing fake values from historical distributions. Both significantly skew load balancing, with misreporting switches attracting over 200% more traffic while staying within a 2–10% deviation. Further analysis in [13] shows such attacks can be tuned via reconnaissance and achieve provable stealth under bounded perturbations, bypassing threshold-based anomaly detectors.

Misreporting led deception also appears in other domains, where diverse defenses have been proposed. SPHINX [14] offers an FSM-based model that captures expected SDN control plane behavior and detects deviations caused by unauthorized rule insertions or topology manipulations. Statistical approaches include Kalman filters [15], z-score filtering [16], and autoencoders for CPS anomaly detection [17]. More advanced methods apply GANs [18], explainable ensembles [19], and hybrid statistical-ML designs [20]. *While promising, most assume adversarial data lies outside historical distributions. Stealthy misreporting instead mimics legitimate behavior, evading outlier-based detection.*

In centrally controlled VR systems, the implications are pronounced. VR requires ultra-low latency and high throughput to sustain immersion, with motion-to-photon (MTP) latency—the time between a user's movement and corresponding visual feedback—typically constrained below 20 ms [21] to avoid dizziness and disorientation. To meet this, compute-intensive tasks such as Visual-Inertial Odometry (VIO) or rendering are increasingly offloaded to edge servers [22]. This creates dependencies on SDN-based routing and switch telemetry, where controller decisions determine task placement. Recent work like XRgo [23] and RemoteVIO [22] show clear performance benefits in power and stability. *Yet none of these systems consider adversarial SDN behavior: a misreporting switch could bias selection toward a compromised server, injecting subtle pose perturbations or spatial drift while still meeting latency budgets—producing degradations invisible to conventional QoS metrics. Although multipath and ML-driven optimizations for VR/AR traffic have been explored [2], [3], [4], the risks of stealthy misreporting in latency-sensitive immersive applications remain underexplored.*

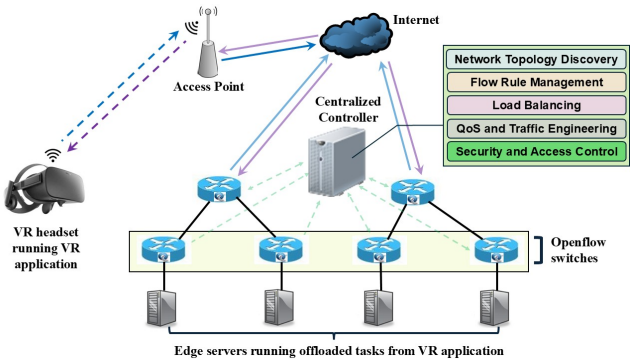


Fig. 1: VR pipeline offloading to remote edge servers using SDN infrastructure

III. System and Threat Model

A. The application pipeline

We consider a VR system deployed over an SDN infrastructure, where compute-intensive components such as rendering or pose estimation are offloaded from the client to remote edge servers. The system runs in fixed-length *workflows* (e.g., 15 seconds), each dynamically routed by the SDN controller based on real-time network conditions. As shown in Fig. 1, the VR client initiates workflows and exchanges data with selected servers through a wireless access network. Edge servers perform the offloaded tasks and return results via SDN-managed switches. These OpenFlow switches forward packets and report per-port statistics (e.g., byte counters) that guide controller decisions. The controller, beyond standard functions, polls switches periodically and assigns each new workflow to the server behind the switch reporting the lowest load, aiming to balance traffic and prevent congestion.

B. Misreporting attack model

While this architecture enables efficient offloading, it relies on trusted switch statistics. An attacker who compromises a server and its switch can exploit this trust to degrade VR QoE. Attack vectors (e.g., malware, Trojans) are beyond our scope. Once compromised, the attacker falsifies port-level load statistics [8] to influence controller decisions, redirecting workflows. Subtle pose manipulations then accumulate across routing intervals, reducing visual coherence and overall VR experience.

We adopt the stealthy misreporting model of [8] as a reproducible adversarial mechanism to stress-test detection. The SDN controller polls edge switches for port-level byte counts and assigns workflows to the switch reporting the lowest load. A compromised switch falsifies reports, with probability φ, replacing its true load with a value from the bottom ρ-th

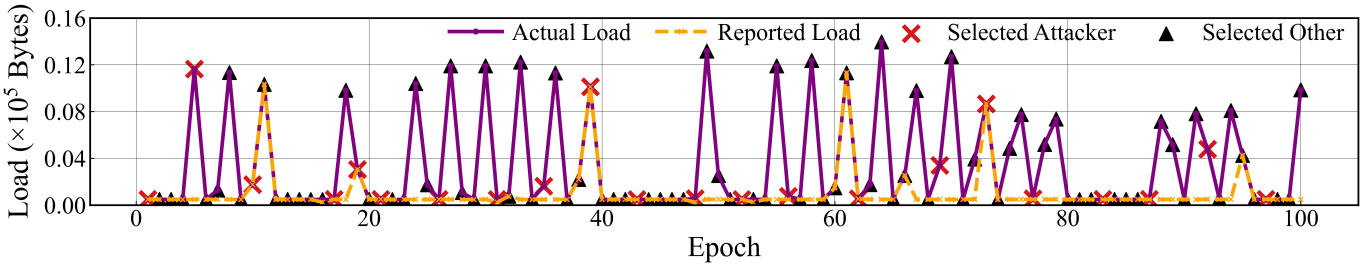


Fig. 2: Reported and actual load values across 100 epochs during a misreporting interval

percentile of its historical loads—blending into natural traffic variation. To attract a target fraction τ of workflows, the attacker sets:

$$\varphi = \frac{\tau - \frac{1}{S}}{(1-\rho)^{S-1} - \frac{1}{S}} \tag{1}$$

where S is the number of switches and ρ controls stealthiness. The attack persists for a bounded epoch window ϵ, after which misreporting stops.

Fig. 2 shows a 100-epoch window from our dataset with $\tau = 0.6$, $\epsilon = 1000$, and $\rho = 0.01$. It compares actual and reported loads at the compromised switch alongside controller decisions. Misreporting occurs when reported load falls below the true load, biasing selection. Red "×" markers denote epochs where the compromised switch was chosen, and black "△" markers indicate other switches. The results show misreporting raises the compromised switch's selection probability while preserving stealth.

Our goal is to design a detection mechanism that identifies malicious switches early in the workflow sequence, preventing further compromise. Mitigation is beyond this paper's scope, though a possible response is provided in Section VI.

C. Quantitative Evaluation of QoE Degradation Under Misreporting: A VR Case Study

To motivate detection, we quantify misreporting's impact on VR QoE. While it may seem counterintuitive that falsified switch statistics alone degrade QoE, our experiments confirm this. Repeated redirection of pose-estimation tasks to a compromised server—via stealthy misreporting—lets the attacker inject controlled noise into pose values before returning them to the client, similar to [24]. These perturbations disrupt the motion-to-photon pipeline, reducing visual coherence and introducing spatial inconsistencies that degrade immersion.

Unlike the Mininet-based setup in [8], our deployment uses edge servers on the Internet-scale FABRIC testbed. To handle higher control-plane latency, we adopt a 2-second polling interval (vs. 1s) for reliable statistics without reducing attack effectiveness. The switching fabric runs Open vSwitch on FABRIC nodes, instrumented with a modified library supporting *normal* and *attack* modes. In normal mode, switches return true per-port byte counters via `dump-ports`; in attack mode, each report follows a Bernoulli trial with frequency φ, and falsified loads are sampled from the bottom ρ-th percentile of historical values to mimic plausible traffic. Attacker parameters—τ (target share), ρ (stealth), ϵ (window), and S (switch count)—are coded in, allowing φ to be computed via Equation 1 and tuned manually. Redirected workflows trigger ILLIXR server modifications that inject noise into pose updates, degrading QoE of client-side VR. This setup forms the adversarial baseline for detection. Impact is measured by comparing estimated trajectories against ground truth using standard error metrics [25]:

- **Absolute Trajectory Error (ATE)**: Evaluates global consistency of predicted poses. After aligning predictions to ground truth via a rigid-body transform, ATE is the RMSE of differences in corresponding poses:

$$\mathbf{e}_i = \hat{\mathbf{T}}_i^{-1}\,\mathbf{S}\,\mathbf{T}_i \tag{2}$$

$$\text{ATE}_{\text{RMSE}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\text{trans}(\mathbf{e}_i)\|^2} \tag{3}$$

  where $\mathbf{T}_i$ is the estimated pose, $\hat{\mathbf{T}}_i$ the ground truth, $\mathbf{S}$ the optimal alignment using Horn's method, and N the number of poses. The translational component of the pose error $\mathbf{e}_i$ is extracted using $\text{trans}(\cdot)$.
- **Relative Pose Error (RPE)**: Captures local motion fidelity via RMSE of relative pose differences over fixed time interval δ:

$$\mathbf{r}_i := \left(\hat{\mathbf{T}}_i^{-1}\hat{\mathbf{T}}_{i+\delta}\right)^{-1}\left(\mathbf{T}_i^{-1}\mathbf{T}_{i+\delta}\right) \tag{4}$$

$$\text{RPE}_{\text{RMSE}}(\delta) = \sqrt{\frac{1}{M}\sum_{i=1}^{M} \|\text{trans}(\mathbf{r}_i)\|^2} \tag{5}$$

  where M is the number of intervals and $\text{trans}(\cdot)$ extracts translation.

To evaluate the impact of compromised pose estimation, we simulate an adversary that intermittently alters edge server pose outputs. Spoofing is applied at four levels: 0% (none), 25% (every fourth pose altered), 50% (alternate poses altered), and 75% (three of four poses altered). These perturbations reduce pose accuracy and rendering quality, producing artifacts such as jitter, reduced immersion, and VR fatigue.

Fig. 3 shows rotational RPE. At 0%, errors stay flat, indicating stable tracking. At 25%, periodic deviations emerge; at 50%, alternating real and corrupted poses cause sharp oscillations and perceptual jitter. At 75%, errors are smoother but consistently high, reflecting orientation drift. Thus, mid-frequency spoofing (50%) yields greater instability than occasional (25%) or continuous (75%) corruption.

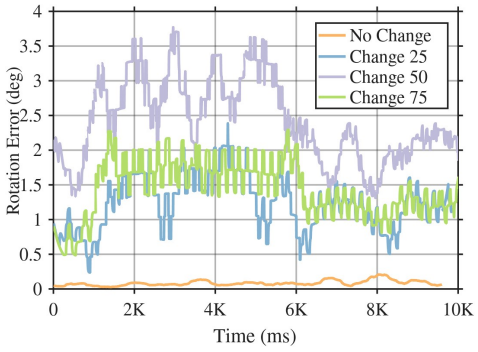
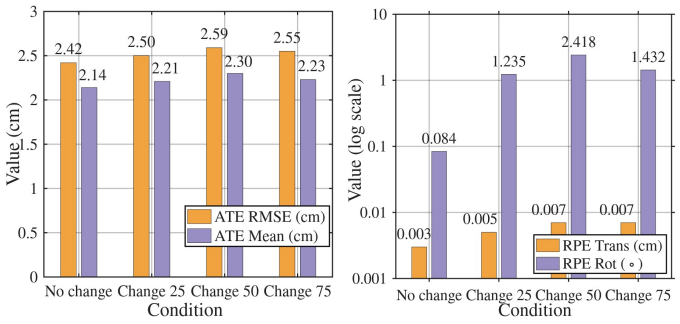


Fig. 3: Smoothed (window = 100) RPE rotation error over time for different levels of pose spoofing



(a) ATE RMSE and mean for client-side trajectories.

(b) Log-scale RPE translation and rotation errors.

Fig. 4: Evaluation metrics under edge perturbation. (a) Trajectory accuracy (ATE). (b) Pose drift behavior (RPE).

Fig. 4a shows ATE increasing with spoofing, with 50% yielding the largest trajectory error. Frequent toggling between correct and corrupted poses accumulates misalignment, showing that inconsistency—not just intensity—harms coherence. Fig. 4b presents translational RPE (log scale): errors rise with spoofing, with 50% and 75% reaching similar levels, while rotational RPE peaks at 50%. Overall, mid-frequency spoofing causes the greatest rotational disruption, and higher spoofing worsens translational accuracy.

IV. Detection Model

To illustrate detection challenges, we show how stealthy misreporting biases controller decisions while evading simple threshold filters. The attacker substitutes true byte counts with values from the lower tail of its history, which appear plausible in isolation. As Fig. 2 shows, with $\tau = 0.60$, $\rho = 0.01$, and $\varphi \approx 0.48$, misreporting can dominate most epochs yet remain statistically believable. This underscores the need for models that exploit temporal patterns and switch-specific baselines rather than point-wise thresholds.

ML offers tools to capture such subtle behaviors across domains. In healthcare, hybrid methods combining isolation forests with supervised classifiers improved anomaly detection in electronic records [26]. In autonomous driving, LIFE exploited sensor correlations to detect spoofing [27], while smart grid defenses used autoencoder–GAN hybrids to classify attacks [28]. IoT systems applied fog-based adaptive learning for real-time detection [29].

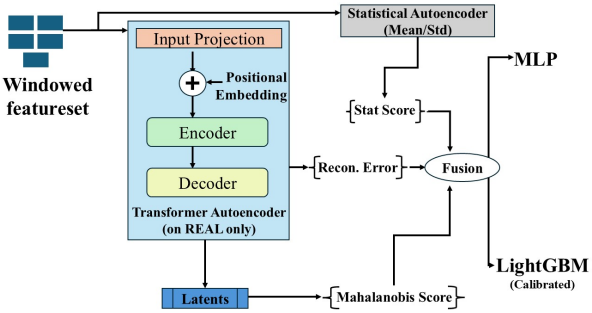


Fig. 5: Pipeline of hybrid detection model

Motivated by these works, we design a hybrid ML framework for misreporting detection in SDN. Since only reported switch values are observable and Fig. 2 shows no per-point distinction between real and fake reports, we extract temporal features to amplify deviations. Features span four groups: *basic load statistics* (raw load, deltas, rolling mean); *distributional indicators* (percentiles, z-score, skewness, kurtosis); *temporal stability* (recent standard deviation, autocorrelation, Mean Absolute Deviation); and *peer context* (load ratios, mean delta across peers, unique counts within a window).

Features are computed per switch over overlapping sliding windows. A window is labeled FAKE if any timestep is misreported, otherwise REAL, enabling detection of localized anomalies. This representation preserves temporal continuity and supports sequential models such as Transformers or LSTMs. As shown in Fig. 5, our pipeline uses a Transformer autoencoder trained on REAL windows to capture benign dynamics and output two unsupervised cues: reconstruction error and Mahalanobis distance. In parallel, a statistical autoencoder produces deviation scores against rolling statistics. These three signals—reconstruction, Mahalanobis, and statistical—form the fused anomaly feature triplet.

Scores are refined using lightweight classifiers—a one-layer MLP and a calibrated LightGBM. LightGBM is calibrated on the validation set with three fused features (reconstruction error, statistical deviation, Mahalanobis distance) via 5-fold cross-validation and sigmoid scaling. This hybrid design adapts to varied attack profiles and improves detection fidelity. Tab. I highlights the limitation of relying only on the Transformer autoencoder: reconstruction error achieves high F1 and accuracy for *REAL* samples, but $F1_{FAKE}$ falls from 0.71 at the 90th percentile to ¡0.5 at the 96th as thresholds tighten. This decline shows stealthy misreporting evades detection under reconstruction loss alone, motivating our hybrid approach that augments latent-space (Mahalanobis) and statistical deviation cues with supervised refinement.

TABLE I: Transformer autoencoder performance

Percentile	Threshold	$F1_{REAL}$	$F1_{FAKE}$	ACC_{REAL}	ACC_{FAKE}
90	0.000049	0.9545	0.7125	0.9215	0.9215
92	0.000076	0.9466	0.6264	0.9065	0.9065
94	0.000109	0.9364	0.5064	0.8873	0.8873
96	0.000146	0.9260	0.3618	0.8674	0.8674
98	0.000197	0.9161	0.1916	0.8480	0.8480

V. Evaluation and Results

A. Implementation

Experiments are run on the FABRIC testbed with compute nodes emulating SDN switches, edge servers, and a VR client. The anomaly detection dataset is generated by running ILLIXR in headless mode with Vulkan Swapchain disabled for CLI compatibility. We deploy Floodlight's STATISTICS-based load balancer, polling per-port byte counts every 2s and routing flows to the switch with the lowest delta. The pool comprises four Open vSwitch instances, each linked to an edge server performing pose estimation, similar to [22].

All components—controller, switches, edge servers, and traffic generator—run on separate FABRIC nodes (Fig. 1). A fifth node generates the ILLIXR-driven VR workload and ICMP background traffic with exponential inter-arrival time. Each epoch is labeled *REAL* or *FAKE* based on misreporting configuration. The workload uses ILLIXR's Offload VIO client, offloading pose data every 15s. Edge servers (2-core CPUs, 4 GB RAM, 100 GB disk, NVIDIA A30 GPUs) execute the VIO pipeline; switches use similar nodes without GPUs. The SDN controller (4-core CPU) runs Floodlight, coordinating flow decisions over FABRIC's internal network.

B. Experimental Configuration

We collect about 48 hours of VR activity data on the FABRIC testbed using ILLIXR. Each session begins with normal load reporting, followed by a period where one switch performs stealthy misreporting. Four sessions are recorded with varying attack parameters ρ, τ, and φ, summarized in Table II.

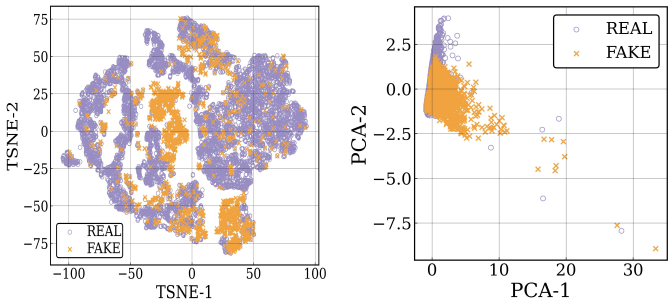
TABLE II: Attack parameter combinations

ρ	τ	φ
0.01	0.48	0.31
0.01	0.29	0.06
0.10	0.48	0.48
0.10	0.29	0.09

Each load-balancing action is followed by a 15s window where the headset offloads pose estimation, so misreporting can influence server selection and QoE. The dataset is shuffled and split into training (60%), validation (20%), and testing (20%), each with 1000 epochs. Misreporting is modeled via Bernoulli sampling to match φ. Models are trained offline on a 24-core CPU, 32 GB RAM, and NVIDIA RTX 2000 Ada GPU, but are lightweight enough for real-time deployment on the SDN controller.

C. Results

1) Classifier Effectiveness: The MLP and calibrated LightGBM achieve high accuracy (0.9659, 0.9661) and AUC (0.9845, 0.9873) as shown in Tab. III. FAKE detection is robust, with precision $[0.9178, 0.9198]$ and recall $[0.8779, 0.8768]$, while REAL precision and recall both exceed 0.97. These results show that shallow classifiers, combined with unsupervised features (reconstruction error, Mahalanobis distance, statistical deviation), provide strong separability. The hybrid design enables accurate, low-latency misreporting detection for real-time SDN use.



(a) t-SNE: partial separation in latent space. (b) PCA: broader FAKE variance on PC1.

Fig. 6: Low-dimensional projections of latent and fused anomaly features. Purple = REAL, Orange = FAKE

TABLE III: Test-set performance of MLP and LightGBM

Metric	MLP	LightGBM
REAL Precision	0.9753	0.9750
REAL Recall	0.9839	0.9843
REAL F1-score	0.9796	0.9796
FAKE Precision	0.9178	0.9198
FAKE Recall	0.8779	0.8768
FAKE F1-score	0.8974	0.8978
Accuracy	0.9659	0.9661
ROC AUC	0.9845	0.9873

2) Latent Embedding Analysis: t-SNE projections of latent features show FAKE samples clustering more tightly and REAL more dispersed (Fig. 6a), indicating that the autoencoder captures some class-relevant structure. However, substantial overlap shows latent embeddings alone lack clear separation. PCA on fused features reveals PC1 explains most variance, with FAKE spanning a wider range than REAL (Fig. 6b), highlighting greater variability in FAKE embeddings and their value for detection.

3) Model Explainability via SHAP: We use SHAP (SHapley Additive exPlanations) to interpret contributions of the fused anomaly features. Fig. 7 shows the global SHAP summary ranking features by impact on FAKE. Reconstruction error (`recon`) dominates, with high values strongly pushing predictions toward FAKE. Mahalanobis distance (`mahal`) provides moderate, consistent support, while statistical deviation (`stat`) has weaker, symmetric influence—high values often favor REAL, low values give mild FAKE support. Overall, `recon` leads, `mahal` supports, and `stat` plays a minor role.

Instance-level attributions illustrate this further. In the true positive case (Fig. 8, left), strong `recon` (+8.54) with `mahal` (+0.45) outweighs weak negative `stat` (-0.11), pushing the logit above the FAKE threshold. In the false negative case (right), all features are positive—`recon` (+2.21), `mahal` (+1.34), `stat` (+0.68)—but the combined signal is insufficient to cross the boundary. This reflects a failure mode where anomaly cues exist but lack strength for detection.

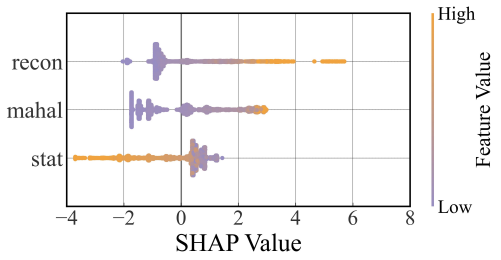


Fig. 7: Global SHAP summary plot showing top contributing features towards classification.

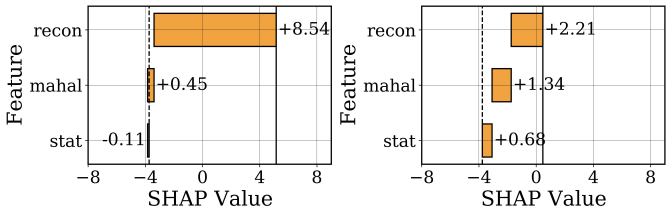


Fig. 8: SHAP waterfall plots from the hybrid model explaining a true positive and false negative prediction. Purple = REAL Orange = FAKE. Dashed = $E[f(x)]$, Solid = $f(x)$.

4) Comparison with Unsupervised Methods: For comparison with a fully unsupervised variant, we replace supervised classifiers with Isolation Forest, One-Class SVM, GMM (Combined and Latent), LOF, and KMeans (Latent). These models use either fused anomaly scores (reconstruction error, statistical deviation, Mahalanobis distance) or Transformer latent features, ensuring fairness against the hybrid baseline.

Tab. IV summarizes results. Isolation Forest performs best among unsupervised models with $\mathrm{F1}_{\mathrm{FAKE}} = 0.6984$ and $\mathrm{AUC} = 0.9339$, capturing high-dimensional deviations but with many false positives from uncalibrated thresholds. One-Class SVM maintains precision but recalls only 32.26% of attacks. GMM (Combined) achieves recall 0.7776 but low precision ($\mathrm{AUC} = 0.6567$), while LOF and KMeans show very low F1. Overall, Isolation Forest detects subtle anomalies but remains unreliable, whereas our Transformer-calibrated hybrid reaches $\mathrm{F1}_{\mathrm{FAKE}} > 0.80$, combining anomaly sensitivity with supervised calibration, interpretability, and real-time viability for SDN defense.

TABLE IV: Performance comparison between the proposed hybrid approach and unsupervised anomaly detectors

Model	Accuracy	Precision_FAKE	Recall_FAKE	F1_FAKE	ROC_AUC
GMM (Combined)	0.5159	0.2283	0.7776	0.3531	0.6567
Isolation Forest	0.9058	0.7651	0.6425	0.6984	0.9399
One-Class SVM	0.8400	0.5495	0.3226	0.4066	0.6778
LOF	0.8249	0.0344	0.0011	0.0022	0.4261
KMeans Latent	0.7198	0.1323	0.1169	0.1241	0.4802
GMM Latent	0.4992	0.1691	0.4977	0.2524	0.4927

5) Deployment Cost and Inference Latency: Tab. V shows that the MLP, statistical autoencoder, Mahalanobis scorer, and LightGBM all meet sub-millisecond budgets. The only exception is the Transformer autoencoder, which records 1.745,s on a resource-constrained FABRIC controller. In realistic deployments, modern edge controllers (e.g., Edgecore AS7326-56X with Intel Xeon D-1518 CPU and 16GB DDR4 RAM) are far more capable, so this latency would be greatly reduced.

TABLE V: Per-sample inference latency across modules

Model	Time (s)
MLP	0.0010
Statistical Autoencoder	0.0050
Mahalanobis	0.0147
Calibrated LightGBM	0.0311
Transformer Autoencoder	1.7451

6) Fusion Component Ablation Study: We evaluate the contribution of reconstruction error, statistical deviation, and Mahalanobis distance, reported in Tab. VI using an ablation study. Using reconstruction alone gives the strongest signal ($\mathrm{F1}_{\mathrm{FAKE}} = 0.880$, $\mathrm{AUC} = 0.980$), while Mahalanobis offers moderate performance ($\mathrm{F1}_{\mathrm{FAKE}} \approx 0.73$, $\mathrm{AUC} > 0.94$). Statistical deviation collapses ($\mathrm{F1}_{\mathrm{FAKE}} = 0$, $\mathrm{AUC} \approx 0.658$). Fusion of all three boosts LightGBM to $\mathrm{F1}_{\mathrm{FAKE}} = 0.898$, $\mathrm{AUC} = 0.987$, with MLP trailing by < 0.02. Across branches, LightGBM slightly outperforms MLP, especially under fusion.

TABLE VI: Ablation study

Model	Precision_FAKE	Recall_FAKE	F1_FAKE	AUC
MLP (Recon)	0.8936	0.8666	0.8799	0.9796
LightGBM (Recon)	0.8988	0.8563	0.8771	0.9786
MLP (Stat)	0.0000	0.0000	0.0000	0.6582
LightGBM (Stat)	0.0000	0.0000	0.0000	0.6568
MLP (Mahalanobis)	0.7274	0.7394	0.7334	0.9503
LightGBM (Mahalanobis)	0.7381	0.7166	0.7272	0.9488
MLP (Fusion)	0.9177	0.8779	0.8974	0.9845
LightGBM (Fusion)	0.9198	0.8768	0.8978	0.9873

7) Cross-Dataset Generalization Benchmark: Tab. VII benchmarks cross-dataset generalization. With longer session intervals, performance drops sharply: in `interval_25`, FAKE F1 drops to 0.21, REAL F1 to 0.17, and AUC to 0.45. `interval_20` recovers REAL F1 (0.79) but leaves FAKE F1 low (0.24) due to class imbalance and attack stealth. In contrast, `epoch_500` remains strong (FAKE F1 = 0.65, REAL F1 = 0.96, AUC = 0.90), showing robustness to attack window variation but sensitivity to session timing.

TABLE VII: Cross-dataset benchmarking metrics

Dataset	FAKE F1	REAL F1	AUC	Accuracy
Dataset_interval_25	0.206571	0.174460	0.451225	0.190834
Dataset_interval_20	0.243271	0.787129	0.562899	0.667727
Dataset_epoch_500	0.646173	0.957318	0.896945	0.923825

8) Sliding Window Sensitivity - Window vs. Stride: Finally, Tab. VIII reports sensitivity to window and stride. Larger windows (15–20) with moderate strides (5–10) give the best results, with FAKE F1 > 0.91 and REAL F1 > 0.98 under settings like (20, 10). LightGBM consistently edges out MLP in AUC and accuracy. Stride = 15 causes slight degradation from under-sampling, confirming that over-aggregation reduces anomaly sensitivity. Overall, performance depends strongly on temporal context length and overlap. All evaluation related codes and data are available through Github [30].

TABLE VIII: Sliding window sensitivity: MLP vs LGB (values reported as MLP/LGB).

Window	Stride	FAKE F1	REAL F1	AUC
5	5	0.774 / 0.751	0.963 / 0.962	0.960 / 0.961
10	5	0.887 / 0.884	0.978 / 0.977	0.978 / 0.978
10	10	0.833 / 0.830	0.965 / 0.966	0.971 / 0.971
15	5	0.892 / 0.899	0.978 / 0.979	0.981 / 0.981
15	10	0.867 / 0.875	0.972 / 0.974	0.978 / 0.979
15	15	0.830 / 0.847	0.965 / 0.969	0.963 / 0.963
20	10	0.911 / 0.916	0.981 / 0.982	0.986 / 0.986
20	15	0.885 / 0.883	0.975 / 0.975	0.982 / 0.982

VI. Conclusions and Future Work

This paper examined stealthy misreporting attacks in SDN-hosted VR systems. Using the FABRIC testbed and ILLIXR framework, we built a realistic SDN-VR pipeline, reproduced misreporting, and demonstrated its impact on pose tracking and scene stability. To counter this, we proposed a hybrid detection framework that combines a Transformer-based autoencoder with statistical and latent-space features, fused into a lightweight supervised classifier, improving F1 by up to 20% over baselines. Future work will explore coordinated multi-switch attacks and application-layer feedback to further strengthen detection in immersive SDN environments.

References

[1] X. Zhang, H. Gan, A. Pal, S. Dey, and S. Debroy, "On balancing latency and quality of edge-native multi-view 3d reconstruction," in *Proceedings of the Eighth ACM/IEEE Symposium on Edge Computing*, SEC '23, (New York, NY, USA), p. 1–13, Association for Computing Machinery, 2024.

[2] M. Xu, Y. Zhou, and Y. Chen, "A monte carlo tree search-based routing scheme with vr video qoe guarantees in sdns," in *2024 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–5, 2024.

[3] F. Zou, Y. Wang, and Y. Liu, "A multipath routing approach for tile-based virtual reality video streaming based on sdn," in *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 560–565, 2021.

[4] K. M. Naguib, I. I. Ibrahim, M. M. Elmessalawy, and A. M. Abdelhaleem, "Optimizing data transmission in 6g software defined networks using deep reinforcement learning for next generation of virtual environments," *Scientific Reports*, vol. 14, no. 1, p. 25695, 2024.

[5] D. Kong, X. Chen, C. Wu, Y. Shen, Z. Zhou, Q. Cheng, X. Liu, M. Yang, Y. Qiu, D. Zhang, and M. K. Khan, "rdefender: A lightweight and robust defense against flow table overflow attacks in sdn," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 9436–9451, 2024.

[6] S. E. Vadakkethil Somanathan Pillai and K. Polimetla, "Mitigating ddos attacks using sdn-based network security measures," in *2024 International Conference on Integrated Circuits and Communication Systems (ICICACS)*, pp. 1–7, 2024.

[7] M. Sinha, P. Bera, and M. Satpathy, "Sdn_guard: An advanced machine learning based defense system against packet injection attacks in sdn," *Procedia Computer Science*, vol. 258, pp. 2490–2499, 2025. International Conference on Machine Learning and Data Engineering.

[8] Q. Burke, P. McDaniel, T. La Porta, M. Yu, and T. He, "Misreporting attacks against load balancers in software-defined networking," *Mobile networks and applications*, vol. 28, no. 4, pp. 1482–1497, 2023.

[9] I. Baldin, A. Nikolich, J. Griffioen, I. I. S. Monga, K.-C. Wang, T. Lehman, and P. Ruth, "FABRIC: A national-scale programmable experimental network infrastructure," *IEEE Internet Computing*, vol. 23, no. 6, pp. 38–47.

[10] M. Huzaifa, R. Desai, S. Grayson, X. Jiang, Y. Jing, J. Lee, F. Lu, Y. Pang, J. Ravichandran, F. Sinclair, B. Tian, H. Yuan, J. Zhang, and S. V. Adve, "Illixr: Enabling end-to-end extended reality research," in *2021 IEEE International Symposium on Workload Characterization (IISWC)*, pp. 24–38, 2021.

[11] M. Chen, T. La Porta, T. Taylor, F. Araujo, and T. Jaeger, "Manipulating openflow link discovery packet forwarding for topology poisoning," CCS '24, (New York, NY, USA), p. 3704–3718, Association for Computing Machinery, 2024.

[12] S. Soltani, M. Shojafar, H. Mostafaei, Z. Pooranian, and R. Tafazolli, "Link latency attack in software-defined networks," in *2021 17th International Conference on Network and Service Management (CNSM)*, pp. 187–193, 2021.

[13] M. Yu, Q. K. Burke, T. F. La Porta, and P. McDaniel, "Stealthy misreporting attacks against load balancing," *IEEE/ACM Transactions on Networking*, vol. 32, no. 4, pp. 3622–3635, 2024.

[14] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "Sphinx: detecting security attacks in software-defined networks.," in *Ndss*, vol. 15, pp. 8–11, 2015.

[15] J. Li, C. Sun, S. Yang, and Q. Su, "Dynamic load altering attack detection based on adaptive fading kalman filter in smart grid," *IET Generation, Transmission & Distribution*, vol. 18, no. 2, pp. 303–313, 2024.

[16] O. A. Lawal, J. Teh, B. Alharbi, and C.-M. Lai, "Data-driven learning-based classification model for mitigating false data injection attacks on dynamic line rating systems," *Sustainable Energy, Grids and Networks*, vol. 38, p. 101347, 2024.

[17] M. Catillo, A. Pecchia, and U. Villano, "Cps-guard: Intrusion detection for cyber-physical systems and iot devices using outlier-aware deep autoencoders," *Computers Security*, vol. 129, p. 103210, 2023.

[18] A. M. Zacaron, D. M. B. Lent, V. G. da Silva Ruffo, L. F. Carvalho, and M. L. Proença Jr, "Generative adversarial network models for anomaly detection in software-defined networks," *Journal of Network and Systems Management*, vol. 32, no. 4, p. 93, 2024.

[19] C. Minh, K. Vermeulen, C. Lefebvre, P. Owezarski, and W. Ritchie, "An explainable-by-design ensemble learning system to detect unknown network attacks," in *2023 19th International Conference on Network and Service Management (CNSM)*, pp. 1–9, 2023.

[20] S. Das and S. Ghosh, "Impact of error rate misreporting on resource allocation in multi-tenant quantum computing and defense," *arXiv preprint arXiv:2504.04285*, 2025.

[21] L. Bassbouss, S. Steglich, and M. Lasak, "High quality 360 video rendering and streaming," *NEM Summit proceedings*, 2016.

[22] Q. Jiang, Y. Pang, W. Sentosa, S. Gao, M. Huzaifa, J. Zhang, J. Perez-Ramirez, D. Das, D. Gonzalez-Aguirre, B. Godfrey, and S. Adve, "Remotevio: Offloading head tracking in an end-to-end xr system," in *Proceedings of the 16th ACM Multimedia Systems Conference*, MMSys '25, (New York, NY, USA), p. 101–112, Association for Computing Machinery, 2025.

[23] S. Gao, J. Liu, Q. Jiang, F. Sinclair, W. Sentosa, B. Godfrey, and S. Adve, "Xrgo: Design and evaluation of rendering offload for low-power extended reality devices," MMSys '25, (New York, NY, USA), p. 124–135, Association for Computing Machinery, 2025.

[24] S. Saha, M. N. Absur, and S. Debroy, "Detection and recovery of adversarial slow-pose drift in offloaded visual-inertial odometry," 2025.

[25] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573–580, 2012.

[26] M. Tabassum, S. Mahmood, A. Bukhari, B. Alshemaimri, A. Daud, and F. Khalique, "Anomaly-based threat detection in smart health using machine learning," *BMC Medical Informatics and Decision Making*, vol. 24, no. 1, p. 347, 2024.

[27] J. Liu and J.-M. Park, ""seeing is not always believing": Detecting perception error attacks against autonomous vehicles," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2209–2223, 2021.

[28] I. Siniosoglou, P. Radoglou-Grammatikis, G. Efstathopoulos, P. Fouliras, and P. Sarigiannidis, "A unified deep learning anomaly detection and classification approach for smart grid environments," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1137–1151, 2021.

[29] S. S. Hameed, A. Selamat, L. Abdul Latiff, S. A. Razak, O. Krejcar, H. Fujita, M. N. Ahmad Sharif, and S. Omatu, "A hybrid lightweight system for early attack detection in the iomt fog," *Sensors*, vol. 21, no. 24, p. 8289, 2021.

[30] GitHub, "Github repository." https://github.com/dissectlab/XR-CNSM2025.git, 2025. Accessed: Sep 21, 2025.