Accelerating Anomaly Detection in Industrial Control Systems Using SmartNICs and DPDK

Sergio Elizalde†, Samia Choueiri†, Ali Mazloum†, Jose Gomez*, Elie Kfoury†, Jorge Crichigno†
†College of Engineering and Computing, University of South Carolina (USC)
*Katz School of Business, Fort Lewis College, USA.
Email: {elizalds, choueiri, amazloum}@email.sc.edu, jagomez@fortlewis.edu, ekfoury@email.sc.edu, jcrichigno@cec.sc.edu

***Abstract*—Industrial Control Systems (ICS) are critical infrastructures that integrate physical processes with programmable logic controllers (PLCs), human–machine interfaces (HMIs), and network communication. Given their dual exposure to cyber and physical threats, continuous monitoring is essential to ensure reliability and safety. A key requirement of ICS is maintaining low latency, as delays can desynchronize control loops and compromise system stability.**

This paper presents a hybrid detection framework that combines sensor-level time-series fault analysis with flow-based network anomaly detection for Modbus/TCP-based ICS. The framework leverages an NVIDIA BlueField-3 SmartNIC to offload machine-learning inference and sequential signal processing directly to the network interface using DPDK. The proposed system employs cumulative sum (CUSUM) and exponentially weighted moving average (EWMA) techniques to extract features for a multilayer perceptron (MLP) classifier, which identifies normal and faulty sensor behavior with high per-device accuracy. Network flow statistics are also analyzed to detect cyberattacks targeting the ICS infrastructure. Experimental results show sub–1 µs average inference latency for binary classification and an average of 5 µs per 32-packet burst on the BlueField-3 SmartNIC.

***Index Terms*—Industrial control systems, Anomaly detection, SmartNICs, Machine learning, Edge computing**

I. Introduction

Industrial Control Systems (ICS) form the operational foundation of modern critical infrastructure, managing processes across sectors such as energy, water treatment, manufacturing, and transportation. These systems combine physical processes with control logic implemented by Programmable Logic Controllers (PLCs) and communication channels connecting field devices and Human–Machine Interfaces (HMIs). As information technology (IT) and operational technology (OT) converge the exposure of ICS networks to cyber threats and operational faults has significantly increased. Because physical faults and cyberattacks often exhibit overlapping characteristics, accurately detecting and distinguishing between them remains a major challenge.

Traditional intrusion detection systems (IDS) in ICS environments typically operate at the network layer and run on general-purpose CPUs. Although effective at identifying protocol misuse and unauthorized communication, these systems are computationally demanding and struggle to meet the stringent latency requirements of real-time control. Frequent context switching, data movement, and memory copying introduce additional overhead, reducing I/O throughput and making conventional IDS unsuitable for low-latency industrial networks. In contrast, SmartNICs integrate packet processing, flow management, and programmable compute resources directly into the data plane, enabling in-line traffic inspection and security enforcement at wire speed [1].

This paper presents a SmartNIC-based hybrid detection framework that unifies physical fault detection with cyber-attack anomaly analysis in Modbus/TCP-based control networks. The proposed system leverages an NVIDIA BlueField-3 SmartNIC to offload machine-learning inference directly to the network interface. A MLP model classifies per-flow and per-sensor features to distinguish between normal and faulty or malicious behavior. For long-term monitoring, techniques such as EWMA and CUSUM were used. This integration minimizes host CPU utilization, reduces data movement overhead, and achieves low-latency anomaly detection by processing and correlating network and sensor data within the SmartNIC itself. The main contributions of this work are as follows:

- A hybrid detection framework that uses per-sensor time-series statistics and per-flow network features, enabling joint detection of physical faults and network attacks in ICS environments.
- An in-network implementation of a lightweight MLP classifier on an NVIDIA BlueField SmartNIC using DPDK.
- A comprehensive experimental evaluation on a publicly available ICS dataset.

The remainder of this paper is structured as follows. Section II introduces the fundamental concepts of industrial control systems (ICS) and SmartNIC technology. Section III reviews related work on data plane machine learning for cybersecurity in ICS. Section IV describes the architecture of the proposed hybrid detection framework. Section V presents the dataset used in this work. Sections VI and VII discuss the two use cases for SmartNIC acceleration. Finally, Section VIII reports the performance benchmarks, and Section IX summarizes the main findings and outlines future research.

This work was supported by the U.S. National Science Foundation under awards 2346726, 2417823, and 2403360.

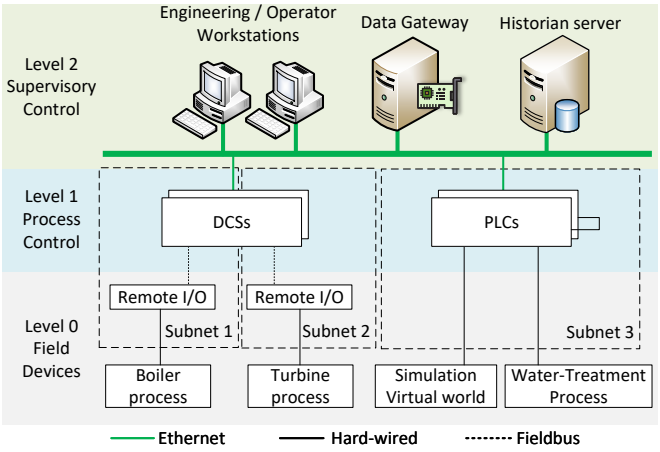


Fig. 1: ICS testbed architecture from [2].

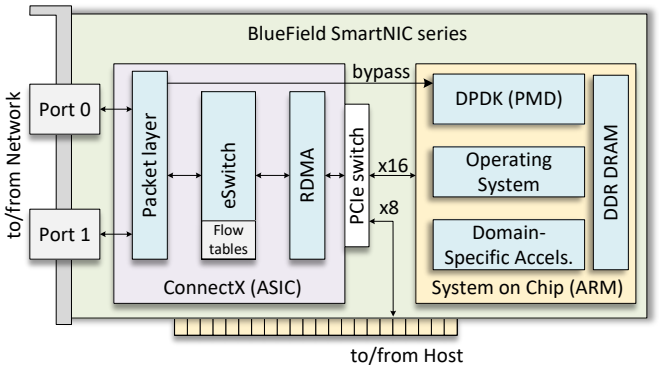


Fig. 2: BlueField SmartNIC architecture integrating a ConnectX ASIC for packet processing and a programmable ARM-based SoC with domain-specific accelerators.

II. Background

A. Industrial Control Systems

ICSs follow a hierarchical architecture that separates supervisory, control, and physical process functions across multiple layers, as illustrated in Fig. 1. At the Level 2, operator and engineering workstations interface with data gateway and historian servers for system management and data aggregation. The Level 1 hosts Programmable Logic Controllers (PLCs) and Distributed Control Systems (DCSs), which coordinate with remote I/O modules to execute real-time control loops. Finally, Level 0 contains the sensors, actuators, and physical processes such as boiler, turbine, and water-treatment systems that directly interact with the environment.

Within this architecture, the data gateway represents a strategic point for deploying SmartNICs. Positioned between the control and supervisory networks, a SmartNIC can perform in-network computation, such as real-time traffic inspection, anomaly detection, or encryption. This offloading reduces host CPU overhead, lowers end-to-end latency, and preserves deterministic communication critical to ICS operations.

B. Overview of SmartNICs and DPDK

SmartNICs are emerging devices that integrate programmable computing resources into the network interface. Originally designed for offloading complex tasks from the host, they are now used for a variety of applications, including security inspection. Figure 2 illustrates the architecture of NVIDIA's BlueField SmartNIC series, which combines a high-performance ConnectX ASIC with an embedded ARM-based System-on-Chip (SoC). The ConnectX ASIC accelerates data plane functions such as packet parsing, flow classification, RDMA, and eSwitch-based switching, supporting hardware-assisted virtualization and flow steering across virtual interfaces. The SoC subsystem runs a lightweight Linux environment and hosts user applications, which can leverage domain-specific accelerators for tasks such as cryptography and compression.

Developers can program the SmartNIC using frameworks such as DPDK or NVIDIA's DOCA SDK to gain direct access to packet queues, flow tables, and eSwitch functions. This capability enables low-latency implementation of security, telemetry, and machine-learning–based analytics directly within the network interface [3].

C. Modbus TCP/IP

Modbus TCP/IP is a communication protocol that enables data exchange between industrial devices such as controllers and sensors over Ethernet networks. Consider Fig. 3. It illustrates the structure of a Modbus TCP/IP Application Data Unit (ADU), which consists of two main components: the MBAP (Modbus Application Protocol) header and the Protocol Data Unit (PDU). The MBAP header includes fields such as the Transaction ID, Protocol ID, Length, and Unit ID. The Transaction ID uniquely identifies each request-response pair, enabling the tracking of messages within a Modbus communication session. The Unit ID corresponds to the PLC identifier. Together, the Transaction ID and Unit ID indicate that a packet belongs to a specific communication flow between a client and a particular PLC. The Function Code defines the operation requested by the client, such as reading or writing data (e.g., function code 03 reads holding registers). The Data field carries the actual sensor information retrieved in bytes.

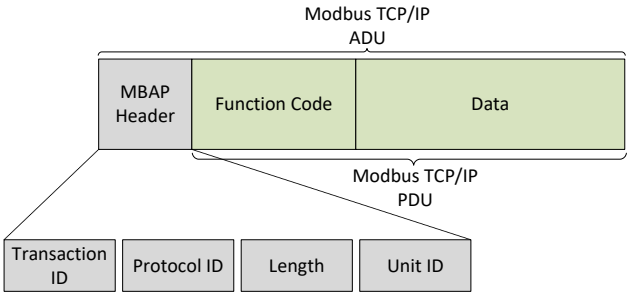


Fig. 3: Modbus packet.

III. Related Work

A. Data Plane Acceleration for ICS

Holik et al. [4] explored P4-based switches as hardware accelerators capable of high-speed, line-rate packet processing through programmable match-action pipelines, while also examining eBPF as a lightweight in-kernel software acceleration framework for flexible packet handling and cybersecurity in OT networks. Similarly, Samson Raj et al. [5] proposed a P4-based in-network security framework for ICSs, showing that programmable switches such as Intel's Tofino can enforce security policies directly in the data plane with minimal delay. AlSabeh et al. [6] proposed a similar scheme for line-rate TLS header inspection. The deterministic behavior of Tofino hardware has been exploited by time-sensitive applications, as noted by Kfoury et al. [7]. Together, these studies highlight the complementary benefits of P4 for deterministic hardware acceleration and eBPF for dynamic software programmability.

Our approach offloads critical network processing tasks to a SmartNIC to leverage P4-style hardware programmability while employing DPDK as a kernel-bypass mechanism for fast, user-space packet processing. This hybrid design aims to merge the deterministic performance of hardware acceleration with the adaptability of software-based processing, resulting in an efficient and scalable system optimized for secure and low-latency network operations.

B. ML for ICS Security and Resilience

Machine Learning (ML) approaches have become increasingly relevant for IDS in ICS due to the evolving complexity and sophistication of cyber threats targeting critical infrastructures. Traditional signature-based or rule-based detection systems are often insufficient in identifying zero-day or stealthy attacks that exploit OT vulnerabilities [8], [9]. ML techniques, both supervised and unsupervised—enable dynamic learning from network and process data, allowing IDS to adaptively distinguish between normal and anomalous behavior [10]. For instance, supervised models like Decision Trees, Random Forests, and Support Vector Machines have been successfully applied to classify attack types in SCADA environments with high accuracy [11].

Previous MLP-based intrusion detection studies show strong detection accuracy but face key performance limitations. Raman et al. [8] used an MLP-CUSUM model for anomaly detection in the SWaT testbed, but its heavy computation and offline processing introduced high latency, limiting real-time use. Similarly, Mesca et al. [9] achieved high precision with a CNN-MLP hybrid but suffered from low recall and software-level delays under high-throughput ICS traffic.

Our work overcomes these issues by implementing the MLP model with DPDK and SmartNIC offloading, enabling real-time packet processing with minimal latency overhead. This hardware-accelerated design supports deterministic, microsecond-scale inference, making ML-based intrusion detection practical for live industrial control environments.

IV. Proposed Framework

A. Overview

The proposed framework integrates fault detection and cyberattack anomaly analysis within a unified SmartNIC-based architecture for Modbus/TCP Industrial Control Systems. Faulty sensor behavior is identified through sequential statistical processing using Exponentially Weighted Moving Average (EWMA) and Cumulative Sum (CUSUM) methods, which capture gradual drifts and abrupt deviations in sensor readings. Cyberattack detection, in contrast, is achieved by parsing Modbus/TCP traffic in the data plane using DPDK to extract per-flow statistical features. Both detection paths employ a lightweight MLP model executing directly on the BlueField-3 SmartNIC, which classifies the extracted feature vectors from sensor and network data. This combination of statistical sequencing and neural inference enables low-latency, in-line identification of both physical and network anomalies while maintaining continuous packet forwarding at wire speed.

B. Sequential Signal Processing

In the first stage, sequential sensor data are analyzed to detect gradual or abrupt deviations indicative of physical or control faults. Two complementary techniques are employed: the EWMA and the CUSUM.

The EWMA technique smooths each sensor signal by weighting recent observations more heavily than older ones. For a time series x_t, the EWMA statistic z_t is computed as

$$z_t = \alpha x_t + (1-\alpha) z_{t-1}, \tag{1}$$

where $\alpha \in (0, 1]$ is a smoothing factor. A larger α increases responsiveness to short-term changes, while a smaller value emphasizes long-term trends. Deviations between x_t and z_t can catch gradual sensor drift or abnormal fluctuations.

The CUSUM method complements EWMA by detecting small but persistent shifts in the process mean. It maintains cumulative sums of positive and negative deviations from a reference value μ_0:

$$C_t^+ = \max\big(0,\, C_{t-1}^+ + (x_t - \mu_0 - k)\big), \tag{2}$$

$$C_t^- = \min\big(0,\, C_{t-1}^- + (x_t - \mu_0 + k)\big), \tag{3}$$

where k is a reference offset controlling sensitivity. While EWMA provides a smoothed representation of trends, CUSUM enables faster detection of small mean shifts, making their combination effective for identifying both gradual and abrupt process faults.

C. DPDK Program

Algorithm 1 outlines the proposed DPDK-based Modbus sensor anomaly detection pipeline implemented on the BlueField-3 SmartNIC. The application continuously polls network interfaces for incoming packets, parsing Modbus/TCP traffic at line rate. Client requests are stored in a lookup table indexed by the transaction and unit identifiers, allowing response messages to be matched with their corresponding

Algorithm 1: Modbus Sensor Anomaly Detection

```
Input: Packets, MLP model
Output: Anomaly alerts
Init: DPDK, sensor tables
while running do
    pkts ← rx_burst();
    foreach p in pkts do
        if Modbus packet (port 502) then
            if Request read then
                store_request(trans_id, unit_id, registers);
            else if Response then
                if matching request found then
                    foreach sensor in response do
                        update_sensor(unit_id, register,
                          value);
                        if window full then
                            pred ←
                              MLP_inference(sensor_features);
                            if anomaly then
                                alert(unit_id, register);
        forward(p);
```

requests. Each Modbus response is decoded to extract register values representing individual sensor readings. For every unique sensor stream, identified by its unit ID and register address, the system maintains a fixed-size window of recent samples. When a window becomes full, statistical features such as minimum, maximum, mean, inter-arrival time, and signal slope are computed directly in the data plane, along with their corresponding CUSUM and EWMA descriptors. These normalized feature vectors are then passed to the MLP inference module, which classifies each window as normal or anomalous. If an anomaly is detected, the SmartNIC raises an alert while continuing to forward packets at wire speed. This design minimizes host involvement, avoids costly context switches and memory copies, and enables real-time inference on streaming sensor data.

V. Dataset

The proposed detection framework was evaluated using the publicly available ICS dataset presented in [12]. This dataset was generated within an extended ICSSIM environment and integrates three synchronized data sources: (i) physical process sensor readings from two PLCs, (ii) system logs from PLCs and HMIs, and (iii) Modbus/TCP network traffic captured in PCAP format. All streams are timestamp-aligned, enabling cross-domain analysis of process-level faults and network-level attacks. As summarized in Table I, the dataset includes Modbus/TCP packets, Syslog messages, and PLC sensor samples collected under three operational categories:

- **Normal operation:** baseline process execution under nominal conditions without injected disturbances.
- **System faults:** five injected fault types—sensor drift, tank leak, overheating (PLC delay), valve malfunction, and memory corruption.

TABLE I: Dataset used for fault and cyber attack detection

Class	Network traffic (pkts)		PLCs
	Modbus/TCP	Syslog	Sensor readings
Normal	2863138	1128018	6534
Fault	1152195	535654	3806
Attack	19600574	59007	N/A
Total	23615907	1722679	10340

- **Cyberattacks:** four network-layer attacks mapped to the MITRE ATT&CK for ICS framework—reconnaissance (T1089), denial-of-service (T0806), man-in-the-middle (T0830, T0831), and replay (T0813).

VI. Use Case 1: Sensor Fault Detection

A. Feature Engineering

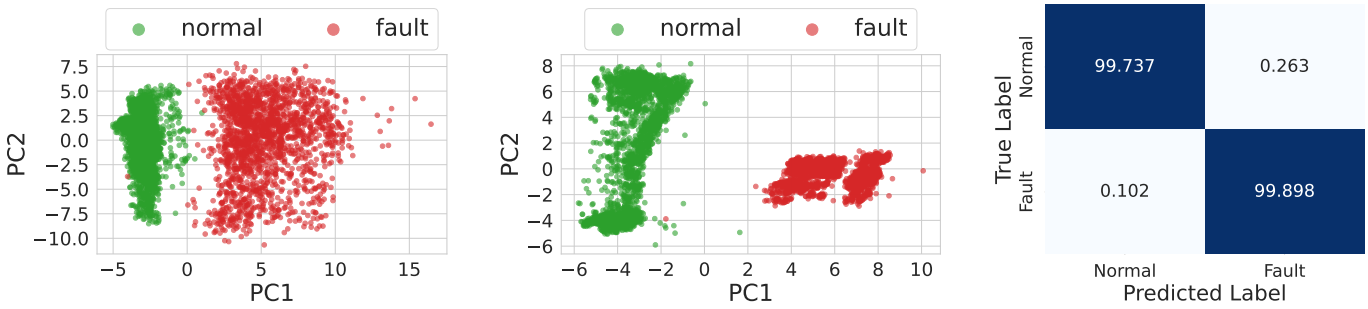
For offline analysis, sensor readings from each PLC are resampled to one-second intervals to ensure temporal consistency across channels (`RESAMPLE_FREQ='1s'`). From the per-second series, four synchronized sequences are derived for each sensor: the raw signal, a rolling z-score (standardized over a 60 s window, `ROLLING_SEC=60`), an EWMA deviation highlighting gradual drifts (`span` $\approx$ 60 s), and a CUSUM trace indicating persistent mean shifts (`CUSUM_DRIFT=0.01`, `CUSUM_THRESHOLD=5`). These signals are aggregated into non-overlapping five-second windows (`WINDOW_SECONDS=5`), from which the mean, standard deviation, and maximum of each measure are extracted.

A subset of the offline feature set is implemented on the SmartNIC to meet in-network processing constraints. Each sensor maintains a buffer of 16 samples (`window_size=16`) from which Exponentially Weighted Moving Average (EWMA, $\alpha = 0.3$) and one-sided CUSUM features are computed. The extractor generates a 16-dimensional feature vector composed of eight raw descriptors (minimum, maximum, mean, inter-arrival statistics, range, and slope) and eight drift descriptors (EWMA and CUSUM means, extrema, and ranges). These feature vectors are then passed to the MLP classifier running on the SmartNIC.

B. Feature Space Separability Analysis

Principal Component Analysis (PCA) is a linear dimensionality reduction technique that projects high-dimensional data onto orthogonal axes (principal components) that capture the maximum variance in the dataset. This transformation enables visualization of complex feature spaces and assessment of class separability by highlighting patterns and clusters in lower-dimensional projections.

Figure 4 presents the feature separability analysis and classification performance of the proposed fault detection framework. Subfigures (a) and (b) show PCA projections of sensor data from PLC 1 and PLC 2, respectively. Both reveal distinct clustering between normal and faulty operational states, with PLC 1 exhibiting more dispersed fault clusters



(a) PLC 1 (b) PLC 2 (c) Confusion matrix

Fig. 4: Feature separability and classification performance. (a)–(b) PCA projections show clear separation between normal and faulty operations for both PLCs. (c) MLP confusion matrix using PCA components for offline MLP training.

due to analog process variations, while PLC 2 shows tighter clusters corresponding to discrete control deviations. This clear separability validates the effectiveness of the extracted features for fault discrimination.

Subfigure (c) shows the confusion matrix of the MLP classifier trained on aggregated data from both PLCs. As expected, the MLP effectively learns and separates the feature space (i.e., the 2D components), achieving over 99% accuracy in distinguishing between fault and normal sensor readings.

C. Accuracy Using Online Model

In this experiment, we evaluated the online feature extraction and model inference process using a subset of CUSUM, EWMA, and standard statistical metrics computed over sliding windows of N packets. Each Modbus-TCP packet capture was parsed to extract sensor-level readings. For each device transaction identifier, non-overlapping windows of $N = 16$ packets were processed to generate 16-dimensional feature vectors representing temporal and behavioral characteristics of the sensor data stream. A total of one million packets were analyzed for both normal and fault datasets to avoid unbalanced dataset.

Table II compares the online classification results using only statistical features against those augmented with CUSUM and EWMA descriptors. The addition of drift-based features slightly improves the overall classification accuracy, increasing the weighted F_1-score from 82.60% to 83.48%. This enhancement reflects a modest gain in robustness without significantly altering the balance between precision and recall across normal and fault classes. Comparing these results to the offline training with a bigger window (i.e., 60 seconds instead of 16 packets) suggests that increasing window size will increase the accuracy at the cost of responsiveness. This trade-off should be carefully tuned depending on the process.

TABLE II: Online classification results using sensor readings

	Statistical only (%)			+CUSUM/EWMA (%)		
Classes	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Normal	93.13	68.2	78.74	93.20	70.04	79.98
Fault	77.90	95.64	85.89	78.90	95.70	86.47
Weighted Average	84.91	83.04	82.60	85.49	83.85	83.48

VII. Use Case 2: Cyber-Attack Detection

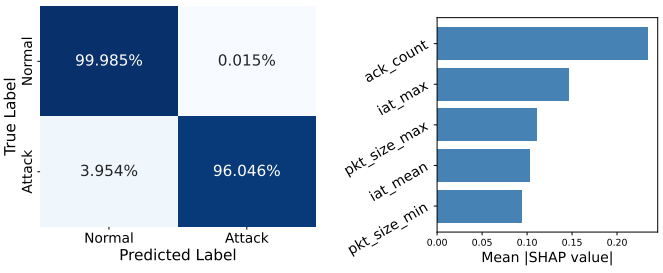
A. Feature Engineering

Cyberattacks manifest their impact primarily at the network layer, exhibiting distinct temporal and behavioral patterns compared to sensor or actuator faults. Consequently, detecting such attacks requires features that capture network-level dynamics rather than physical process deviations.

In the data plane, raw packet captures are converted into flow-level representations that summarize TCP traffic behavior over time. Each flow, defined by its source and destination IP addresses and ports, is divided into fixed-size, non-overlapping windows of sixteen packets. For each window, statistical features are extracted, including packet size metrics (minimum, maximum, and mean), inter-arrival time (IAT) statistics (minimum, maximum, and mean), and TCP control flag counts (SYN and ACK). This set of features provide discriminative information for identifying abnormal behavior.

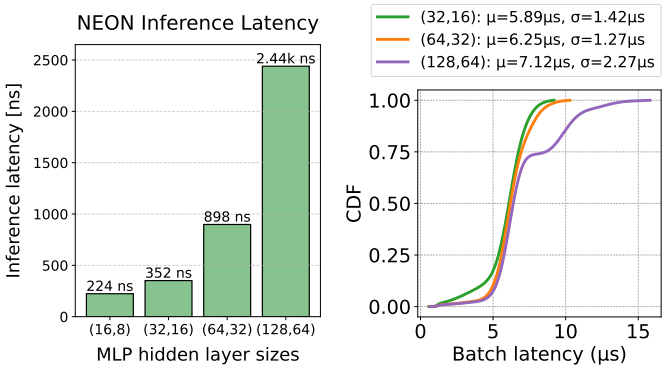
B. Accuracy Using Online Model

The different attacks vectors were labeled as anomalies. To ensure balanced representation of normal and attack traffic, the first one million packets from each PCAP trace were used for training. Figure 5 presents the performance of the MLP-based



(a) Confusion matrix (b) Top-5 feature importance

Fig. 5: Results for cyberattack detection showing model performance and feature contribution.



(a) Inference latency (b) Packet processing latency

Fig. 6: Performance results of the DPDK-based implementation on BlueField-3.

detection model. The confusion matrix in Figure 5a shows a high classification accuracy, achieving 99.98% true negative and 96.05% true positive rates for normal and attack classes, respectively. The SHAP feature importance in Figure 5b highlights `ack_count`, `iat_max`, and `pkt_size_max` as the most influential features, indicating that acknowledgment frequency and timing irregularities are key indicators of anomalous behavior in network flows.

VIII. Performance Results on SmartNICs

To evaluate runtime performance, Pktgen-DPDK was used to replay pre-captured Modbus-TCP traffic traces toward the DPDK application running on the SmartNIC. The DPDK application was configured with a burst size of 32 packets. High-resolution timestamp counters were employed to measure both inference latency and end-to-end packet-processing latency. The trained MLP models were offloaded to the SmartNIC by converting their learned parameters into C header files containing the layer weights and biases. To further enhance computational efficiency, a vectorized ARM-NEON implementation was employed to accelerate matrix–multiply operations.

Figure 6a shows the average per-inference latency for different MLP model sizes, denoted as (H_1, H_2) hidden layer dimensions. The latency increases with model size, reflecting the higher computational cost of deeper networks. Nonetheless, all configurations achieve sub–3 µs mean inference times.

To assess the overall data plane performance, we also measured the packet-processing latency under sustained Modbus-TCP traffic (See Fig. 6b). The cumulative distribution functions (CDFs) show sub-10 μs latency for all configurations, with mean values increasing slightly as model complexity grows. For the $(16, 8)$ model, the latency curve overlaps with that of $(32, 16)$ and is therefore omitted for clarity. These results demonstrate that hardware-level vectorization enables low-latency, high-throughput inference within the SmartNIC data plane, confirming that SmartNICs are well suited for deployment in ICS environments with strict latency and determinism requirements.

IX. Conclusion and Future Work

This paper presents an exploration of a new synthetic dataset [12] for fault detection and network-level cyberattacks in ICS. The proposed pipeline applies CUSUM and EWMA techniques to process sensor data and train an MLP model that classifies sensor time series as normal or faulty with high per-device accuracy. Network flow statistics are also analyzed to detect potential cyberattacks, enabling per-flow anomaly detection. Feature selection and design were guided by the goal of supporting efficient in-line processing without increasing computational cost or degrading performance. Online results show approximately 90% accuracy for fault detection and 99% for cyberattack detection. On the BlueField-3 SmartNIC, vectorized per-core inference achieves sub–1 µs average inference latency. Future work includes integrating the sensor fault detection with the cyber-attack detection on the same DPDK pipeline, and enabling multi-classification for more granular reports.

Acknowledgment

The work was supported by the National Science Foundation, under awards 2403360 and 2400729.

References

[1] S. Elizalde, A. AlSabeh, A. Mazloum, S. Choueiri, E. Kfoury, J. Gomez, and J. Crichigno, "A survey on security applications with smartnics: Taxonomy, implementations, challenges, and future trends," *Journal of Network and Computer Applications*, p. 104257, 2025.

[2] H.-K. Shin, W. Lee, J.-H. Yun, and B.-G. Min, "Two ICS Security Datasets and Anomaly Detection Contest on the HIL-Based Augmented ICS Testbed," in *Cyber Security Experimentation and Test Workshop*, ser. CSET '21, 2021, p. 36–40.

[3] A. Mazloum, A. AlSabeh, E. Kfoury, and J. Crichigno, "Domain name security inspection at line rate: TLS SNI extraction in the data plane using P4 and DPDK," in *IEEE 2025 International Conference on Communications (ICC)*, 2025.

[4] F. Holik, M. M. Cook, X. Li, A. A. Shah, and D. Pezaros, "Programmable data planes for increased digital resilience in OT networks," *IEEE Communications Magazine*, 2025.

[5] R. Samson Raj and D. Jin, "Leveraging Data Plane Programmability Towards a Policy-driven In-Network Security Framework for Industrial Control Systems," in *Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems*, 2025, pp. 152–163.

[6] A. AlSabeh, A. Mazloum, E. Kfoury, J. Crichigno, and H. S. Berry, "Enabling line-rate TLS SNI inspection in P4 programmable data planes," in *IEEE/IFIP 2025 Network Operations and Management Symposium (NOMS)*, 2025.

[7] E. Kfoury, J. Crichigno, and E. Bou-Harb, "Offloading media traffic to programmable data plane switches," in *IEEE 2020 International Conference on Communications (ICC)*, 2020.

[8] G. R. MR, N. Somu, and A. P. Mathur, "A multilayer perceptron model for anomaly detection in water treatment plants," *International Journal of Critical Infrastructure Protection*, vol. 31, p. 100393, 2020.

[9] M. MESCA and G. Stamatescu, "Intelligent Intrusion Detection System for Cybersecurity of Water Utilities," *2025 RoEduNet Conference: Networking in Education and Research*, 2025.

[10] S. Mubarak, M. Habaebi, M. Islam, F. Rahman, and M. Tahir, "Anomaly Detection in ICS Datasets with Machine Learning Algorithms." *Computer Systems Science & Engineering*, vol. 37, no. 1, 2021.

[11] S. Mubarak, M. H. Habaebi, M. R. Islam, A. Balla, M. Tahir, A. Elsheikh, and F. Suliman, "Industrial datasets with ICS testbed and attack detection using machine learning techniques," *Intell. Autom. Soft Comput*, vol. 31, no. 3, pp. 1345–1360, 2022.

[12] D. Sudyana, "ICS Dataset," Sep. 2025. [Online]. Available: https://doi.org/10.5281/zenodo.17165850