International Journal of Technology

http://ijtech.eng.ui.ac.id

Research Article

Resource-Efficient Deep Packet Inspection and Dashboard for Activity and Energy Sensing Supporting Eco-Friendly ICT Infrastructure

Ruki Harwahyu[1], **Abdul Fikih Kurnia**[1], **Muhammad Suryanegara**[1,*]

[1]Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Depok, Indonesia, 16424

[*]Corresponding author: m.suryanegara@ui.ac.id; Telp.: +62217270078; Fax.: +62217270078

Abstract: An organization typically has various enterprise apps and Internet of Things (IoT) systems. The usages of these systems are typically well reflected by the packets transmitted to the network. This paper presents a unique approach to strive for energy savings in organizations by exploiting fact observables in networks. DPI is employed to dissect passing network packets and predict their protocols. Data Plan Development Kit (DPDK) is adopted to push the hardware limit and speed up the inspection. A web-based customizable dashboard is incorporated to allow human observation according to the characteristics of the organization. Several API endpoints are provided to extend the functionalities, such as ML/AI integration. An energy-sensing model is proposed to measure the energy usage or generation by various connected systems, such as IoTs, smart ACs, and solar panel controllers. The results demonstrate that the working prototype improves processing efficiency, enabling the system to handle large volumes of data up to 10GB, achieving an average packet processing efficiency of 99.991%. The system can also accurately identify various protocols, mapping the cybersecurity risks and anomalies, with an average packet loss rate of only 0.83%. The standardized UAT confirms the system's usability, reliability, and robust security. The findings of this study are expected to provide a practical, reliable, efficient, and user-friendly network monitoring solution while contributing to the development of open-source and flexible network traffic monitoring and green technology.

Keywords: Data Plane Development Kit (DPDK); Deep Packet Inspection (DPI); Energy efficiency; Green networking; nDPI

1. Introduction

The drive toward growth usually forces an increase in carbon emissions (Zagloel et al., 2023). This inevitability can be observed in many fields. For example, in the education industry, each student requires certain amounts of resources. Online/remote learning is no exception because it also uses computing and networking resources in the server (Shirota et al., 2019)(Feliana et al., 2023). However, digital resources are generally perceived as greener solutions (Domingos et al., 2024)(Kusrini et al., 2023). Being greener is often managed by digitally representing something that was originally physical (Whulanza, 2023). This can be translated as a digital twin or industry 4.0 in general (Rehman et al., 2023).

https://doi.org/10.14716/ijtech.v16i6.7655
Received February 2025; Revised March 2025; Accepted April 2025

In today's connected world, machines/things are also receiving and sending data (Jha et al., 2022). The opportunity brought by being connected is enormous (Whulanza, 2023), including the chance to be greener, as discussed earlier. However, the other side of this coin is the risk of increasing complexity due to the huge amount and various types of data traversing the network (Hananto et al., 2024). An adequate resource to monitor network data needs to be provided.

This study seeks a niche opportunity to strive for security and energy awareness based on meticulous observation of network activity. Deep packet inspection (DPI), a more capable form of network monitoring, is employed to dissect passing network packets and predict their protocols. DPDK is adopted to push the limit of the hardware of the network adapter and speed up the inspection. Then, a web-based dashboard is incorporated for human observation. Several API endpoints are also provided to extend the functionalities, such as allowing ML/AI to learn and suggest certain insights. These contributions form an awareness tool that can be useful for continuous effort toward network and energy efficiency. Several motivating use cases of the tool are as follows:

— Seeking to reduce energy usage of networking devices. In a production network, inefficiency can be caused by (i) useless or redundant transmissions, (ii) excessive control packets, (iii) unnecessary packet encapsulation, and (iv) unintended packets. Case (i) can be caused by improper configuration in lower layers of the network. Case (ii) can be caused by improper configuration in various layers or unexpected changes in the network topology. Case (iii) can be caused by outdated software or hardware (including firmware). Case (iv) may result from various types of cyber-attacks. A poorly monitored and guarded network becomes an additional attack vector to disrupt digital and physical assets, which may lead to higher energy consumption.

— Business activity baselining and anomaly detection (Yeremia Nikanor Nugroho, 2023) Business processes may involve humans and machines/IoT which send/receive data to/from the network. Characterizing their traffic during normal operation is important so that anomalies can be realized when they occur. Anomalies can be caused by multiple sources, such as

  - Human or system error during the operation
  - Insider or outsider cyber attacks
  - Behaviour mismatch/incompatibility of new system (from update, upgrade, or newly procured) that was unidentified during preparation
  - Decisions on special occasions (which are not well identified and prepared in low-maturity organizations)

  Anomalies may pose various impacts to the organization, including decreasing the efficiency of the system (such as a smart room controller that cannot turn the cooling off and excessively chill the room) and increasing the carbon footprint.

— Ensuring that the invested modern/smart solutions work as expected. Many modern solutions have their own separate dashboard, such as building management, energy generation and usage, general room and ambient appliances IoT, plumbing/water and sewage, industrial control systems (ICS), and supervisory control and data acquisition (SCADA). In addition, various information systems exist within organizations. Depending on the system, unexpected behaviour may be realized from their traffic in the network. Early detection can facilitate rapid mitigation and response, thereby preventing further losses.

— Continuous improvement and auditing. Modern systems and network infrastructures are typically active 24/7 and may grow (capacity and topology wise) along with the organization. Audit and continuous improvement are both best practices for a mature organization.

In the realm of logging and monitoring which produce large amounts of data, ML/AI has also shown promising advancement to perform automatic decision-making (Zhu et al., 2023). However, improving efficiency is often (and ideally is) a long-term commitment (Chairina and Tjahjadi, 2023). With various dynamics in the organization's business process, the decision or intervention fed by network traffic monitoring should be done widely, accountably, and below the authority/responsibility of certain officers (needless to say: human). This niche requires a visual representation of the data (Park et al., 2021), which is most conveniently provided by a customizable dashboard in our contribution.

As of 2024, there are 5.35 billion internet users globally, constituting 66.2% of the world's population (Kemp, 2024). This number represents an increase of 1.8% over the past year, with 97 million new users accessing the internet for the first time in 2023 (Kemp, 2024). Additionally, the projected number of IoT devices worldwide is expected to reach 32.1 billion by 2030 (Vailshery, 2024). This unprecedented growth has resulted in vast data volumes and increasingly dynamic online activities. Traditional tools, such as firewalls or intrusion detection systems, are limited to examining data at the network (Layer 3) or transport (Layer 4) layers of the OSI model, analyzing only packet headers (Ghosh and Senthilrajan, 2019). In contrast, DPI operates up to the Application Layer (Layer 7), enabling more comprehensive analyses, including examining the content of transmitted data (Özbay and Dalkılıç, 2023).

Given the rising diversity and complexity of network traffic, ISPs and telecommunication companies increasingly rely on DPI for traffic monitoring and anomaly detection. However, commercial DPI solutions are often closed-source and expensive, making them inaccessible for broader use (Raza et al., 2024). Therefore, an open-source DPI system capable of effectively monitoring, analyzing, and managing network data traffic is needed. Such systems should be scalable, cost effective, and accessible to a wider community.

Based on previous work on the comparison of accuracy among various DPI toolkits or libraries (Çelebi et al., 2023), the highest accuracy score was achieved by nDPI (91 points), an open-source toolkit capable of recognizing a wide range of protocols and widely used by the developer community. It was followed by PACE (82 points), a commercial toolkit focusing on in-depth protocol analysis but limited in accessibility due to its licensed nature. UPC MLA (79 points), designed to detect protocols through a machine learning approach, offers flexibility but requires significant training data. Finally, Libprotoident (78 points) is a lightweight toolkit suitable for simple protocol detection but limited in the number of protocols it can recognize.

Therefore, the DPI toolkit used in this work is nDPI (ntop DPI), developed by ntop for high-accuracy data packet analysis and available as open-source software. One of the main drawbacks of nDPI is its relatively slow performance when processing large volumes of data. Many studies have discussed ways to improve the performance of nDPI, such as employing multithreading approaches (Deri, 2021), algorithm optimization (Sarhan et al., 2024), or specialized hardware usage (Deri et al., 2024). However, these methods are often insufficient to handle increasing workloads. As a more effective approach, integrating nDPI with the DPDK has been proposed. DPDK is a technology designed to enhance the performance of data packet processing by reducing network operations overhead. It enables faster and more efficient packet processing through low-level programming

and direct hardware access. Therefore, a system that is easy to use, understand, and implement efficiently and quickly is needed.

Several key challenges are addressed in this study. First, the emergence of new Internet protocols, such as Quick UDP Internet Connections (QUIC) and HTTP/3, presents challenges in effective detection and analysis (Joarder and Fung, 2024). Without accurate detection, these protocols can conceal harmful or unlawful content, thereby compromising data security and integrity. Second, with increasing data volumes, efficient DPI systems are essential for handling large-scale data without compromising speed or accuracy, particularly in identifying complex threats hidden within packet payloads (Raza et al., 2024). Third, the accurate identification of network protocols, especially in critical sectors such as banking, is vital for ensuring network security and mitigating cyber threats (Ali, 2019). Finally, existing open-source solutions often lack comprehensive capabilities for detecting and monitoring rapidly evolving or complex protocols. Addressing these gaps is crucial for creating accessible and effective network monitoring systems (Sapp et al., 2021).

The scope of this study is outlined as follows. It focuses on protocols supported by nDPI, which includes over 400 types defined in its library. It evaluates the DPI system performance under various network traffic conditions and the adoption of emerging protocols. The technical implementation focuses on DPI using nDPI and DPDK, along with the development of an intuitive web-based interface. Legal and policy considerations are beyond the scope of this study. System implementation and testing are limited to controlled environments, requiring further validation in production settings. Additionally, the system is tailored for specific organizational or institutional contexts, necessitating additional adaptation for broader applicability.

In summary, this paper makes the following technical contributions:

— Contribute a new mechanism to the open-source nDPI library that enables real-time output display within the backend.

— Integration of nDPI as an engine capable of sending analyzed data to a web-based dashboard for enhanced accessibility and visualization.

— Creation of a user-friendly web dashboard equipped with various features, including AI-powered notification analysis, to assist users in effectively understanding and analyzing network traffic.

— Providing an open-source DPI monitoring solution that is easy to use, accessible to the public, and designed for independent development and customization.

2. Methods

2.1 Deep Packet Inspection (DPI)

Network deep packet inspection (DPI) was introduced to address the limitations of manual port-based approaches in network traffic classification, leveraging payload analysis to detect protocols and hidden threats within network traffic (Ramey, 2024). Various DPI deployments are illustrated in Figure 1. Data capture and analysis can be performed separately or integrated into a single workflow. The data collected from the network can be stored in files for further analysis. The main principles of DPI are pattern matching and event analysis. Since the algorithm should be executed against a large number of packets, the basic low-level method that has been matured over decades (used by routers, switches, and NICs) is chosen over ML/AI-based high-level method.

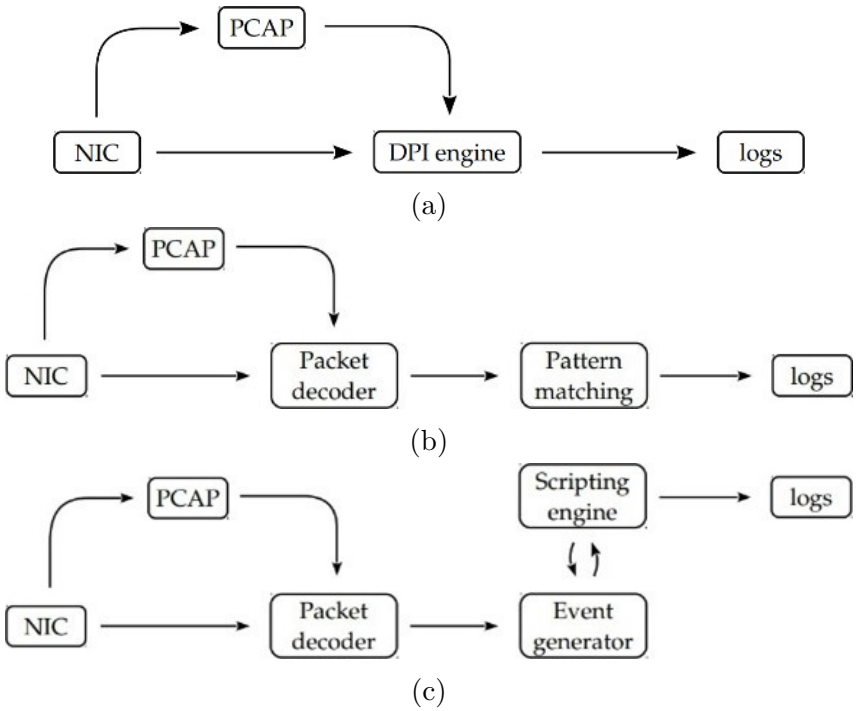

Figure 1 Illustration of DPI deployment: (a) direct analysis, (b) pattern matching, and (c) event analysis

Patterns identified through byte sequences or regular expressions are often easy to detect, making this technique quite popular. However, challenges arise when searching for patterns that cannot be described using regular expressions, such as when data must be decoded before the patterns can be identified. For example, identifying expired SSL certificates from HTTPS connections with a specific list of IP addresses is a task that is too complex to accomplish with regular expressions.

The event-based analysis architecture is employed for more complex detection. In this approach, packets are processed into events, which are analyzed using predefined scripts. Once the packets are decoded, the script engine processes the generated events. This method allows for the application of more complex processing algorithms and the addition of new DPI-related features within these scripts.

This method utilizes algorithms in computer programs to replace the pattern-matching component. These algorithms can be either stateful or stateless. Stateful algorithms can store data or states to be used in subsequent events, whereas stateless algorithms react directly to the events (Harwahyu et al., 2024).

The nDPI library provides the main functionality of DPI (Özbay and Dalkılıç, 2023). It was inspired by the OpenDPI software, which was GPL-licensed but no longer maintained. nDPI is compatible with Linux, Windows, MacOS, and BSD operating systems. It can identify over 400 application protocols and report associated metadata, such as TLS certificates, browser names, and encryption details (Deri, 2021).

DPDK is adopted herein to increase computing efficiency. DPDK is a software framework designed to accelerate the development and performance of the data plane in network systems (Belkhiri et al., 2022). Developed by Intel, DPDK provides APIs for processing data packets directly through software, enhancing network systems' overall performance and efficiency. The adoption of DPDK in the proposed approach optimizes the processing speed, aiming to reduce the unnecessary bottleneck of the additional computation. According to an existing study (Attawna et al., 2023), DPDK implementation significantly

increases the data rate of general processing in software-defined networking switches, with up to 8 folds increases under higher load compared to the baseline implementation found in OpenVSwitch. (Vorbrodt, 2023) presented a performance comparison study of the top three implementation strategies found in the market for network packet processing, namely, the standard Linux kernel, using input-output library, and DPDK. The study found that DPDK yields the highest performance, which is more than 733% compared to the second best, which is the using input-output library.

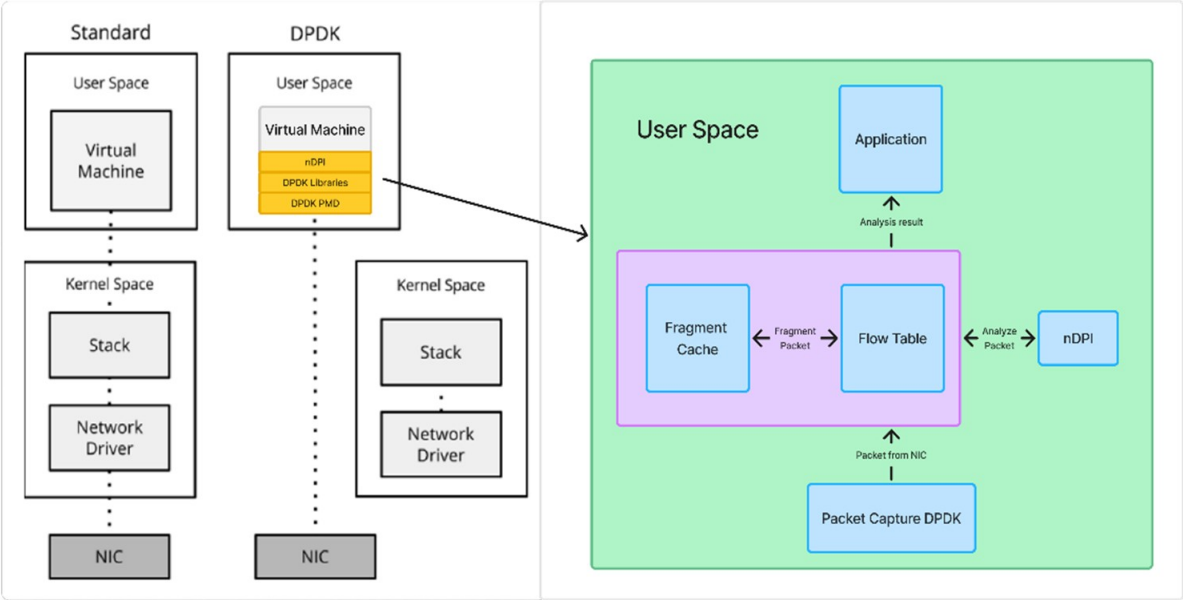

Figure 2 Comparison of default/standard processing and processing using DPDK (left) and the usage of DPDK in the proposed system (right)

DPDK bypasses the OS kernel, allowing applications to interact directly with hardware, as illustrated in Figure 2. This approach eliminates the additional overhead typically incurred when data packets pass through the kernel. Furthermore, DPDK is optimized for multicore architecture, enabling applications to execute tasks in parallel and significantly improve overall performance (Yen, 2021).

Figure 3 shows the protocol identification process performed by nDPI. This process begins with the capture of incoming data packets, which are then classified based on their signatures or the information contained within the packets.

2.2 Website-based interface

A web-based graphical user interface (GUI) is developed using Vue.js and WebSocket protocol to ensure real-timeness. ZeroMQ (ZMQ) is utilized as the communication technology between the backend and frontend. ZMQ offers performance benefit, which has been demonstrated in 5G/cellular controller (Jonnavithula et al., 2024), industrial IoT (Sarasola et al., 2024), enterprise microservice architecture (Ibrahim, 2024), machine learning nodes (Wang et al., 2024), and so on. The usage of ZeroMQ to handle internal communication between elements in the proposed system is illustrated in Figure 4. ZeroMQ facilitates fast and efficient data exchange through various communication patterns, such as publish-subscribe, request-reply, and push-pull. By leveraging ZeroMQ, this monitoring system enables direct real-time data transmission between the backend components and the web interface, ensuring high performance and low latency without overhead.

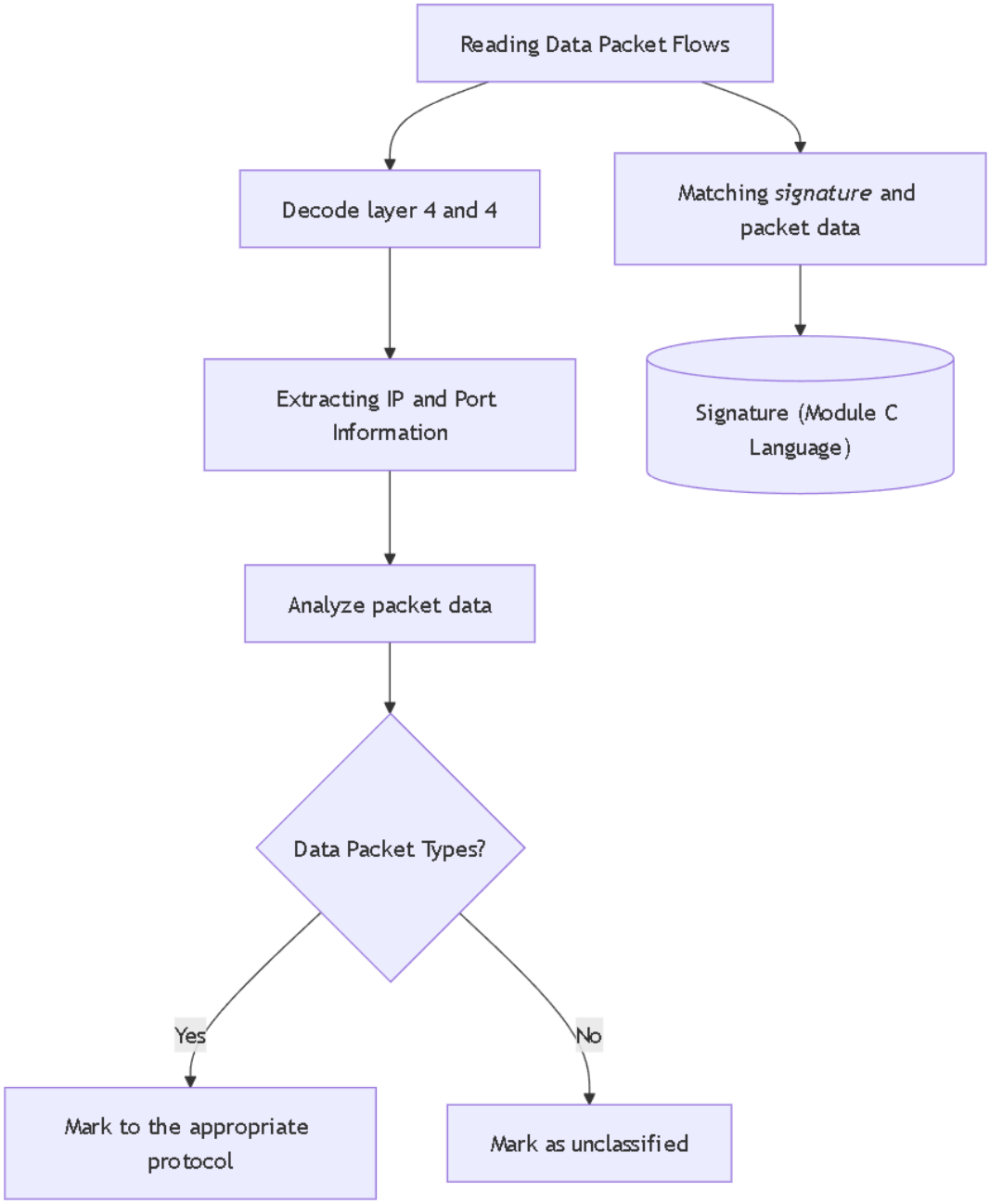

Figure 3 Proposed protocol detection algorithm

This web-based system comprises several interconnected components/microservices as follows:

- The frontend is responsible for presenting information to users through intuitive visualizations. It displays real-time monitoring data using graphs, tables, and interactive widgets, providing a dynamic and responsive user experience.
- The backend acts as a bridge between the user interface and the data processing system. It handles data reception, processing, and real-time transmission. Additionally, the server temporarily stores data during monitoring sessions to ensure system responsiveness.
- Data handler, which stores collected data in a database to ensure efficient and organized access. The database allows permanent data storage, ensuring availability

even after monitoring sessions are completed or the application is closed.

- Application programming interface (API) endpoints, which provide flexible data access between the user interface and the backend system. They enable efficient data retrieval for real-time communication and non-critical data access that does not require immediate updates.
- The monitoring engine performs a deep analysis of network traffic to detect protocols or specific patterns. This component is essential for identifying and classifying potential threats or anomalies. The analysis is conducted using algorithms designed for high efficiency and accuracy.
- Notification system that delivers immediate alerts to users when anomalies or issues are detected in the monitored data. Notifications can be sent through various channels, such as email or instant messaging, to ensure that users can take necessary actions promptly.

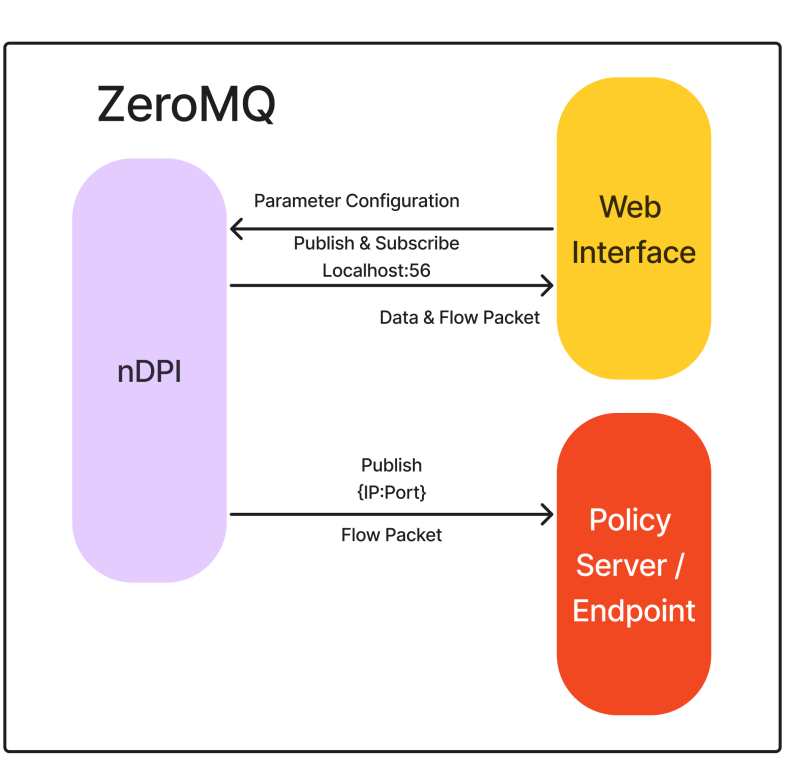

Figure 4 Communication architecture using ZeroMQ

Since various components above communicate using ZMQ, we devise two communication strategies using ZMQ, as illustrated in Figure 5. Depending on the deployment strategy, communication may happen inside a host machine or between different hosts (i.e., host machine and backend API endpoints). Strategies are used to tradeoff between simplicity (simple to maintain and less error prone), security, and performance (less bottleneck).

The developed DPI system and its dashboard allow the detection of network traffic. Normally, the energy usage is approximated for each transmitted packet (Azari et al., 2022). In fact, network traffic correlates with the activities of humans and things that are assisted by computer-based communications (Jia et al., 2019). Therefore, a model is proposed herein to approximate their energy usage.

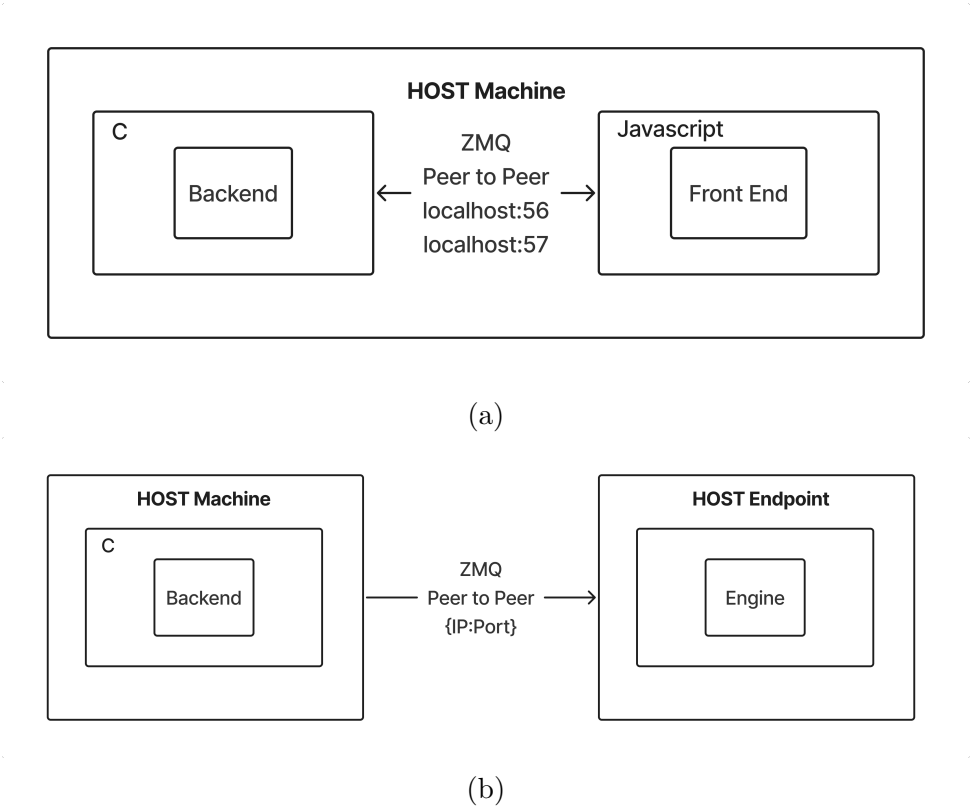

Figure 5 Proposed communication architecture (a) between hosts and (b) between the host and other endpoints

2.3 The energy sensing model

The total energy usage of the involved networking devices is estimated in the first step. Second, the energy usage of each end node that transmits and/or receives the data is estimated. These end nodes include personal smartphones/wearables, computers/workstations, casual room IoTs, industrial IoTs, and building management IoTs. The generated packets may affect the workings (and thus, energy usage) of other non-connected appliances/machineries depending on their communication purpose. For example, in an eco-friendly smart campus, according to the number of registered students, the room air conditioner is turned on to prepare for a class. To autonomously execute this smart action, several packets are communicated via the network. Third, the energy usage of these smart appliances is predicted based on the packet detected in the network.

The network is generalized as an undirected graph G(n,e) with n nodes/vertices and e edges between two random nodes, defined as follows:

$$G = (N, E) \tag{1}$$

where $N = \{n_1, n_2, \ldots, n_s\}$ is the set of nodes, and $E \subseteq N \times N$ with $|E| = e$. From this graph, an adjacency matrix can be established as follows:

$$A_{i,j} = \begin{cases} 1, \text{if there is an edge between node i and j} \\ 0, \text{otherwise} \end{cases} \tag{2}$$

In today's common single-site enterprise network, transmission distance has a negligible impact on total energy consumption. Hence, the propagation delay caused by the edge length in our graph is not included here.

The probability that node i will start transmission at time t is denoted as $S_i(t)$. The probability of node i to transmit a packet with length (in time) l is denoted as $L_i(l)$. Let $T_i(t)$ denote the probability that node i is in the transmitting state at time t. This can be expressed as follows:

$$T_i(t) = \sum_{t=-\infty}^{t} S_i(t') \sum_{l=1}^{\infty} L_i(l) I(t' + l > t) \tag{3}$$

where $I\,(t' + l > t)$ is an indicator function that equals 1 if the packet transmitted at time t' is still being sent at time t, i.e., if I covers the current time t, and 0 otherwise. Note that if node i starts a transmission at time t', it remains in the transmitting state during packet l. In a continuous-time model, this can be rewritten using an integral as follows:

$$T_i(t) = \int_{t'=-\infty}^{t} S_i(t') \sum_{l=1}^{n} L_i(l) I(t' + l > t) dl dt' \tag{4}$$

Node i consumes as much as $e_{tx,i}$ energy unit per time when in the transmitting state. Hence, the energy usage expected by node i due to transmission is

$$E_{tx,i}\,(t_1, t_2) = \int_{t_1}^{t_2} e_{tx,i} T_i(t) dt \tag{5}$$

The base energy usage of each device i connected to the network (end or intermediary device) is generalized as $e_{ON,i}$ for each unit of time. Energy used by the device while powered on. In the real usage of the proposed DPI system, known devices can be added to the database along with their energy profile. In this mathematical model, let E_{ON} be the total base energy usage during [t1,t2], defined as follows:

$$E_{ON}\,(t_1, t_2) = \sum_{i=1}^{n} \int_{t_1}^{t_2} e_{ON,i} P_{ON,i}(t) dt \tag{6}$$

Each transmitted packet traverses the network to reach its destination(s). Within an organization, the destination node can be a gateway router or an end node, such as a computer or Internet of Things (IoT) node. In our graph G, the sending and receiving nodes may be separated by 1 (directly connected) or more edges (via an intermediary node such as an access point or networking switch). When a packet arrives at the immediate intermediary device, it is forwarded to 1 (unicast) or more (multicast or broadcast) nodes depending on the packet type.

Each node i may generate a different number of packets for each packet type, each of which represents different off-network activities or applications with a distinct energy usage profile. It also represents a different set of destination nodes. For example, an IoT controller can send packets to one smart AC to adjust the room temperature (impacting the AC's energy usage) or send packets to all smart ACs in the building to query the current ambient condition (small or no impact on the AC's energy usage). Let $Y_i(y)$ denote the probability that the packet sent by node i is a type of y. Each packet of type y from node i is generalized to be sent to end nodes specified by a set $D_{i,y}$ where

$$E_{i,y} \subseteq \{1, 2, \ldots, N\} \setminus \{i\} \tag{7}$$

In this case, node j is assumed to consume $e_{rx,j}$ energy per time unit in its receiving state. Over a period [t1,t2], it consumes energy to receive a packet

$$E_{rx,j}\,(t_1, t_2) = \begin{cases} \int_{t_1}^{t_2} T_i(t) Y_i(y) e_{rx,j} dt, if j \in D_{i,y}, \\ 0, \text{otherwise.} \end{cases} \tag{8}$$

Each type y packet represents off-network or application energy usage of $e_{app,y}$ per time unit. Note that $e_{app,y}$ can be negative to represent energy generation (e.g., solar panel installations). In real-world usage, $e_{app,y}$ can be measured or set based on the baseline of each application, updated with a certain calibration procedure, or learnt by ML algorithms. The expected application energy usage that can be sensed via the packets generated by node i over a period of [t1,t2] is

$$E_{app,i}(t_1, t_2) = \int_{t_1}^{t_2} S_i(t) \sum_{y_i} Y_i(y) e_{app,y} dt \tag{9}$$

Finally, we need to count the energy usage by intermediary nodes that receive and forward packets. With adjacency matrix A, the set of intermediary nodes M(i,j) along a path from node i to node j can be defined as follows:

$$M(i,j) = \{k \in N \setminus \{i,j\} \,|\exists \text{ a path}(i \to k \to j) : A_{i,k} = A_{k,j} = 1, \text{or no path exist}\} \tag{10}$$

We can express the energy used by all intermediary nodes between nodes i and j as

$$E_{fwd,k}\,(t_1, t_2) = \begin{cases} \int_{t_1}^{t_2} t(e_{rx,k} + e_{tx,k})dt, \text{if } k \in M(i,j), \\ 0, \text{otherwise.} \end{cases} \tag{11}$$

Up until this point, we have obtained the expressions for energy used for powering on the nodes, E_{ON}, for the sender node to transmit packet, E_{tx}, for the intermediary nodes to forward the packet, E_{fwd}, for the destination node to receive the packet, E_{rx}, and for the off-network applications controlled by the packet, E_{app}. Finally, the total energy used by the whole network and the connected application for duration [t1,t2] can be summarized as follows:

$$E_{tot}(t_1, t_2) = \int_{t_1}^{t_2} \sum_{i=1}^{n} E_{ON,i}(t_1, t_2) + E_{tx,i}(t_1, t_2) + E_{app,i}(t_1, t_2) + \sum_{\forall j \in N \setminus i} \left(E_{rx,j}(t_1, t_2) + \sum_{\forall k \in N \setminus \{i,j\}} E_{fwd,k}(t_1, t_2) \right) \tag{12}$$

3. Results and Discussion

The development of the network traffic monitoring system based on DPI using the DPDK framework resulted in the development of a prototype called DeepNet as the PoC. The system comprises two primary modules: the nDPI module and the DPDK module. The nDPI module is responsible for classifying protocols within network traffic, leveraging its ability to identify over 400 protocols, including encrypted and unencrypted traffic. Meanwhile, the DPDK module accelerates packet processing by bypassing the OS kernel and directly interacting with the NIC, enabling high-speed and low-latency operations. The system is further enhanced by a user-friendly web dashboard for real-time visualization of network traffic statistics, protocol details, and anomaly detection, making it a comprehensive tool for network monitoring and management.

The integration results of the nDPI module into the network traffic monitoring system include several adjustments to ensure seamless functionality. The nDPI module was successfully integrated into the developed network traffic monitoring system, enabling accurate and efficient protocol classification. The use of DPDK to accelerate packet processing in the network traffic monitoring system has been successfully implemented using

DPDK version 21. The results of this implementation include the creation of compiled files optimized with DPDK. These files were built using a custom Makefile specifically designed for DPDK integration to ensure efficient packet processing within the system. Additionally, the module successfully transmits these data to the website and other endpoints using ZMQ. The DPDK framework will be implemented in the subsequent stages to accelerate packet processing in network traffic, further enhancing the developed network traffic monitoring system. Packet processing in network traffic has been successfully implemented using DPDK, with configurations set to 128 huge and using 4 threads, concluding with the DPDK state operating on the specified port.

The developed **dashboard** displays general information about the monitored network traffic, while the events page provides detailed information on events occurring within the network traffic. The Configuration page allows users to configure the network traffic monitoring system. Figure 6 shows the code structure of the developed website.

On the dashboard page, information is displayed in alignment with the user requirements carefully designed and outlined in the previous sections. This page serves as the primary interface for monitoring network traffic, offering users a clear and comprehensive view of key metrics and activities. Several functionalities have been developed and implemented to ensure that the dashboard effectively meets its intended purpose. As detailed in earlier chapters, these functionalities provide real-time insights and actionable data to enhance user experience and support decision-making. The information presented on the dashboard includes memory, traffic, detected, protocol, and risk statistics.

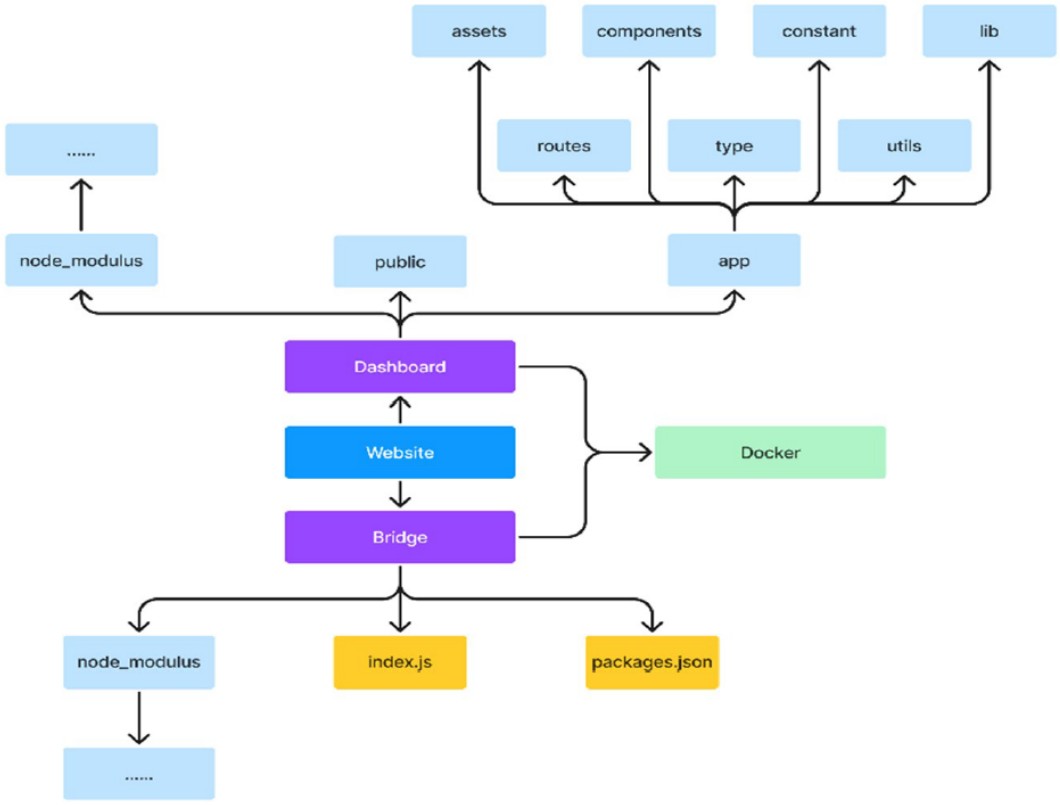

Figure 6 Functional structure of the proposed tool

The Events page serves as a detailed interface that provides comprehensive information about events occurring within the network traffic. It is designed to provide users with an in-depth view of network activities, enabling better understanding and analysis of traffic patterns and anomalies. This page is equipped with several functionalities to ensure

accurate event tracking and real-time data presentation. This page displays information about events occurring in the network traffic. This includes a list of events and their detailed information. The system demonstrates advanced network detection by providing in-depth insights into events and traffic details, a capability not previously developed in an open-source framework.

The Configuration page is designed to allow users to configure various aspects of the network traffic monitoring system, ensuring that it operates effectively and aligns with their specific needs. The Configuration page displays information related to the configuration of the network traffic monitoring system. The page provides options to customize the monitoring system according to user requirements. The configuration settings can be adjusted, and the system can accept these parameters. One notable feature is the ability to configure notifications, which can be tailored to user needs and enhanced with AI-driven analysis, allowing alerts to be sent directly to the user's email address. The proposed system is evaluated under the following experimental setup:

3.1 Protocol Detection

This testing was conducted to determine whether the nDPI module can accurately detect protocols within the network traffic. The testing method involved browsing various websites and applications using a search engine. The test output is a Boolean array denoting the detection status (success/fail). The test was performed using 791,615 packets consisting of 40,686 flows. Each flow represents a 2-way reciprocal connection-oriented communication. This represents 132 common application types. Additionally, the test continued with the detection of 30 and 220 specific flows from the Skype protocol with 30 flows and Microsoft with 220 flows. From this test, the nDPI module can successfully detect all protocols within network traffic. The module can identify HTTP, MDNS (webRTC), SSDP, TLS, QUIC, and even specific protocols, such as WhatsApp, YouTube, Microsoft, Zoom, and Reddit.

Table 1 shows an example of specific protocol detection, namely, Skype and Microsoft, which also commonly appears in local networks, enterprise networks, and the Internet. The test was done 5 times to measure the consistency. The results show that the accuracy is 96.67% and 99.55% for Skype and Microsoft, respectively.

Table 1 Result of detection testing, example for Skype and Microsoft flows

Iteration	Skype Flow	Microsoft Flow
1	29	221
2	27	223
3	30	221
4	29	220
5	30	220

3.2 Anomaly Detection

The performance of anomaly detection was evaluated to determine whether the network traffic monitoring system can successfully identify anomalies within the network traffic. The results of anomaly detection testing using sample anomalous traffic created with tools available in Linux and Kali Linux are presented below.

Table 2 Result of anomaly detection test

No.	Test type	Description and command to generate the anomaly	Result
1	Packet Fragmentation	Sent fragmented packets using: sudo hping3 -I eth0 -f -p 80 -S 192.168.1.102	Detected
2	Malformed Packet	Sent malformed packets using: sudo ping -s 65507 192.168.1.102	Detected
3	Uncommon TLS ALPN	Sent TLS handshake with uncommon ALPN using: openssl s_client -connect testurl.com:443 -alpn http/2	Detected
4	Crawler/ Bot Traffic	Sent requests with uncommon user agent using: wget –header="User-Agent: Googlebot" http://testurl.com	Detected
5	TLS Certificate Expire	Accessed website with expired TLS certificate using: echo — openssl s_client -connect 192.168.1.3:8006 2¿/dev/null — openssl x509 -noout -dates	Detected

3.3 Network performance test

First, throughput testing was conducted to determine the capability of the system to process packets in network traffic. The test used Iperf to send data across the network with varying bandwidths. The testing duration was set to 60 s, and the packet size was 1400 bytes. The results are shown in Figure 7 and Table 3.

Table 3 Comparison of the throughput of DeepNet and the host's default implementation

File Size	DeepNet (pps)	Host (pps)	Comparison (DeepNet/Host)(%)
100 Kb	18.93	18.94	99.95%
1 Mb	83.89	83.95	99.92%
10 Mb	717.78	717.78	100.00%
100 Mb	7450.00	7450.00	100.00%
1 Gb	58830.00	58830.00	99.73%
10 Gb	69910.00	69910.00	99.91%

This result reveals two distinct trends based on the file size. For small file sizes ranging from 100 KB to 10 MB, the achieved throughput was relatively low and nearly identical for both DeepNet and the default implementation, with values ranging between 18.93 and 717.78 ups. This is attributed to the high protocol overhead ratio compared to the payload for small file sizes, which limits the number of packets processed per second. At this stage, the performance of both systems was nearly identical, indicating that the DeepNet optimization had no significant impact.

For large file sizes ranging from 100 MB to 10 GB, the throughput increased significantly, peaking at approximately 69,910 ups for DeepNet and 69,970 ups for the System. With larger file sizes, the protocol overhead became relatively smaller compared to the payload, allowing the systems to utilize network resources more efficiently. The stable increase in throughput for larger files demonstrates the system's ability to handle data efficiently without major hardware or software bottlenecks. Table 3 presents the detailed values of the throughput quality assessment results.

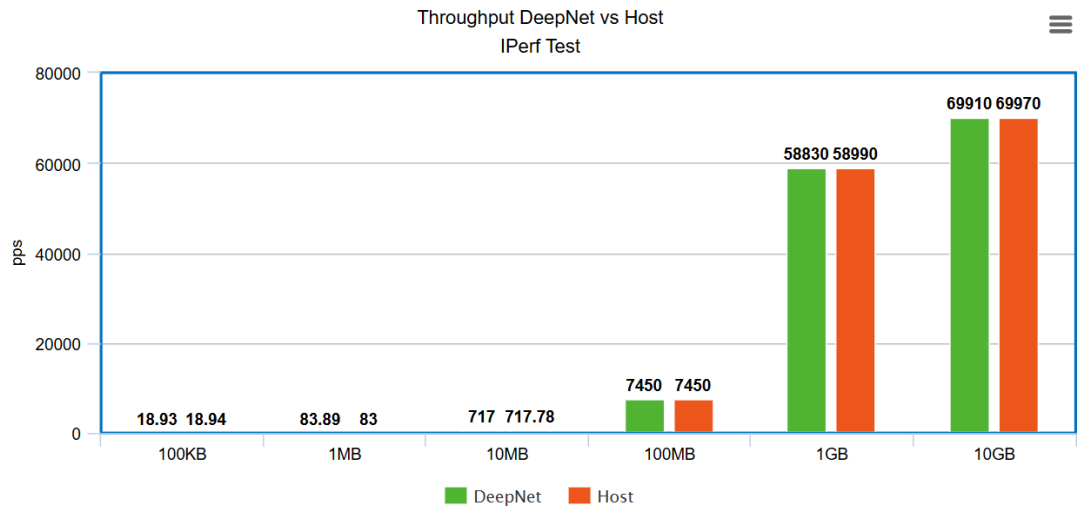

Figure 7 Comparison of throughput (pps) under various analysis loads

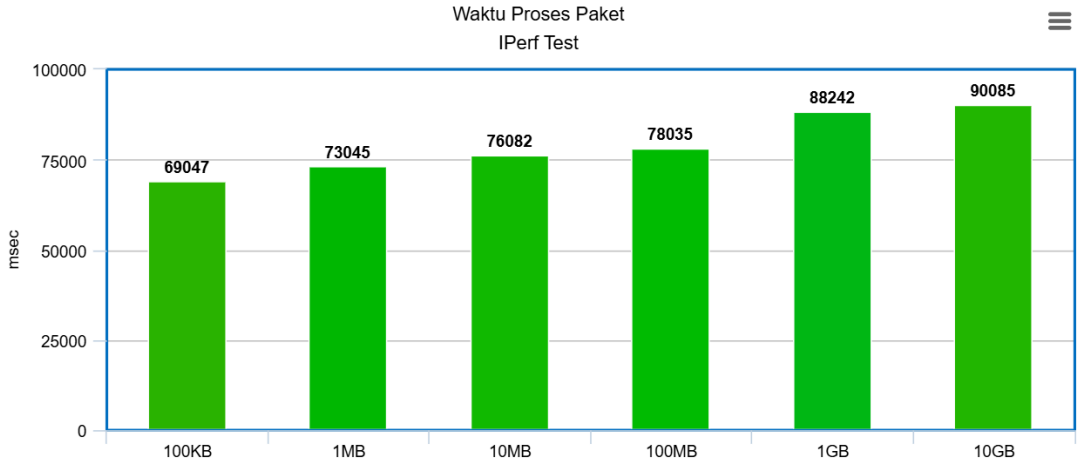

Figure 8 Packet processing time under various load sizes

Second, we conducted packet processing time testing to evaluate the speed of DeepNet, which leverages nDPI-based deep packet inspection (DPI) technology, in processing packets within network traffic. The test involved data sizes ranging from 100 KB to 10 GB to observe trends in packet processing time based on varying traffic volumes. The testing duration was set to 60 s, and the packet size was 1400 bytes. The results indicated that packet processing time increased with larger data sizes; however, the system maintained efficiency for larger data sizes (1 GB to 10 GB), with only a marginal increase in processing time compared to the preceding sizes. The initial processing times for smaller data sizes were likely influenced by protocol overhead, which should be addressed in future system optimizations. Figure 8 and Table 4 show that the proposed system performs well, particularly in processing large data sizes. The nDPI technology effectively contributes to the detection of deep network protocols without causing significant increases in processing time for large datasets. However, the processing time for smaller data sizes highlights opportunities for optimization, particularly in reducing the protocol overhead impact.

Third, packet loss testing aimed to measure the percentage of lost packets during DeepNet data processing. Packet loss indicates that the processing is slower than the load arrival rate, packets are queued, and one or more packets will be dropped (i.e., lost and not processed) if it is waiting too long in the queue. The test was conducted by transmitting data of various sizes, ranging from 100 KB to 10 GB, to evaluate the

system's performance consistency in handling network traffic. The testing duration was set to 60 s, and the packet size was 1400 bytes. According to the results shown in Table 4, the system demonstrates stable performance with a very low packet loss rate (0.001%) for large data sizes (1 GB to 10 GB). However, for smaller data sizes (100 KB), the packet loss rate was relatively high, reaching 4.42%, possibly due to initial processing overhead or resource limitations during small packet transmissions. Overall, the results indicate that the DeepNet maintains excellent reliability for large data sizes, with nearly zero packet loss. The significant reduction in packet loss for larger data highlights the efficiency of the algorithms and the system's stability in handling high-intensity traffic, making DeepNet suitable for large-scale network environments.

Table 4 Packet loss test results

Variation	Packets Processed	Total Packets	Packet loss(%)
100 Kb	1307	1378	4.42%
1 Mb	6128	6161	0.53%
10 Mb	54551	54591	0.073%
100 Mb	536485	536511	0.0048%
1 Gb	5191183	5191217	0.00065%
10 Gb	6297546	6297569	0.00036%

Table 5 Memory usage test results

Variation	Memory Capacity (MB)	Memory Peak (MB)	Memory Peak (%)
100 Kb	4000	26.66	0.66%
1 Mb	4000	33.23	0.83%
10 Mb	4000	99.56	2.4%
100 Mb	4000	762.56	19.06%
1 Gb	4000	4000	100%
10 Gb	4000	4000	100%

Lastly, the memory usage testing was aimed at evaluating the extent of the DeepNet's memory resources while processing network traffic with varying data sizes. Table 5 presents the memory usage results based on data sizes ranging from 100 KB to 10 GB, with a testing duration of 60 s and a packet size of 1400 bytes. The results indicate that memory usage increased proportionally with the processing size of the data. For data sizes up to 100 MB, memory usage remained below the system's maximum capacity of 4 GB. However, the system reached its maximum memory capacity for larger data sizes of 1 GB and 10 GB, highlighting its inability to store all data in memory at once. To address this limitation, the system implemented a data eviction mechanism to ensure the continuous processing of new data. The findings also demonstrate the efficiency of the system in using memory, as memory usage for smaller data sizes (100 KB to 10 MB) was relatively low, remaining below 20%. For larger data sizes (1 GB to 10 GB), memory usage reached 10%, suggesting the need for further optimization to improve resource efficiency in handling high-volume traffic, such as data compression techniques or batch processing.

3.4 User Acceptance Test

User Acceptance Testing (UAT) was conducted to evaluate the quality and performance of the DeepNet from the user's perspective. This test involved users representing network owners or system administrators who would use DeepNet in a production environment. This UAT was also aligned with the ISO/IEC 25010:2011 standard, which defines the quality characteristics of software to ensure that the DeepNet meets the user's requirements (Imran et al., 2024). The following ISO/IEC 25010:2011 aspects were applied during testing:

Table 6 Result of anomaly detection test

Page	Feature	Test Description	Status
System	Accessibility Check	Ensures the dashboard is fully accessible, including navigation via keyboard and screen readers.	Pass
System	Browser Compatibility	Tests the dashboard's compatibility across multiple browsers (e.g., Chrome, Firefox, Safari).	Pass
System	Security Testing	Ensures configurations are secure, preventing unauthorized changes to system settings.	Pass
Dashboard	nDPI Memory Statistics	Displays memory usage by detected protocols using nDPI technology.	Pass
Dashboard	Traffic Statistics	Shows network traffic statistics by detected protocol categories.	Pass
Dashboard	Detected Protocols	Lists detected protocols in the network traffic.	Pass
Dashboard	Protocol Statistics	Provides detailed protocol statistics, including packet counts and data volume.	Pass
Dashboard	Risk Statistics	Displays identified risk statistics based on network traffic patterns.	Pass
Event	Monitoring Flow Event	Monitors network flows in real-time to detect anomalies or suspicious patterns.	Pass
Configuration	On/Off Connection Backend	Enables or disables backend connection to the central server.	Pass
Configuration	On/Off Notification	Enables or disables notifications to the administrator.	Pass
Configuration	On/Off Analyst AI	Enables or disables AI-based anomaly detection modules.	Pass
Configuration	Email Notifications	Configures email settings for receiving activity or anomaly notifications.	Pass
Configuration	IP and Port Endpoint	Configures the IP address and port endpoint for data transmission to the server.	Pass
Configuration	Data Collection Interval	Sets the data collection interval for efficient monitoring.	Pass

Functionality: The test evaluated the system's core functionality, such as the ability to detect and display protocol statistics, risk statistics, and network anomalies on the dashboard. Key functionalities, such as configuration management (e.g., enabling/disabling backend connections, notifications, and AI-based anomaly detection), were tested to confirm that they functioned as expected. This test also indicates that the provided functionality can be used with the proposed energy sensing model to approximate the energy usage or generation by various connected network systems.

Performance Efficiency: Real-time monitoring and data processing performance was tested to ensure that the system could handle large volumes of network traffic without lag or data loss. The data collection interval configuration was validated to ensure optimal performance under various network conditions.

Usability: We assessed the dashboard and configuration interface for ease of use and clarity of the presented information. The test participants were asked to assess the intuitiveness of navigation and the presentation of key metrics, such as traffic statistics and detected protocols.

Reliability: Tests were conducted to evaluate the system's stability during high-traffic scenarios and its ability to recover gracefully from unexpected interruptions. The network risks and anomalies were consistently monitored under varying conditions to ensure reliability.

Compatibility: Compatibility tests were performed to confirm that the web interface operated correctly on different browsers and devices. The system's integration with existing network setups, including configuring endpoints and email notifications, was also verified.

The UAT results are shown in Table 6. This table shows that the provided features in all provided pages are working as expected and are within the acceptable test criteria (pass).

4. Conclusions

The DPI-based system with DPDK is effective in performing network monitoring. DPDK enhances data processing efficiency with optimal speed and high accuracy. Under various loads tested during the evaluation, the system can reach an average throughput of 99.991%. The system can accurately identify various protocols, including associated risks and anomalies. The error detection rate remains under 4%. The detection result is sufficient to be used with the proposed energy sensing model to approximate the energy usage or generation by various networked systems. By leveraging nDPI, the system overcomes the limitations of open-source resources, providing a more affordable solution for detecting complex protocols with an average packet loss rate of 0.83%. The system also passes the UAT based on the ISO/IEC 25010:2011 standard. This study presents a new possibility to observe the activities of various systems in an organization to increase efficiency, such as avoiding anomalies that may lead to energy inefficiency. Future research will assess the accuracy of the energy model and equip it with prediction capabilities that may be achieved using ML approaches.

Acknowledgements

This work was supported by the Ministry of Education, Culture, Research, and Technology, Directorate General of Higher Education, Research, and Technology, Directorate

of Institutional Affairs, and LPDP through the Indonesia – Nanyang Technological University Singapore Institute of Research for Sustainability and Innovation Mandatory Innovative Productive Research Program (INSPIRASI) for 2023–2028 (Batch II) under the contract No. 2966/E4/AL.04/2024 and 307/PKS/ WRIII/UI/2024, dated May 2, 2024, between the Directorate General of Higher Education, Research, and Technology and Universitas Indonesia.

Author Contributions

Ruki Harwahyu: conceptualization, methodology, supervision, manuscript drafting, formal analysis, validation. Abdul Fikih Kurnia: data curation, investigation/experiment, validation. Muhammad Suryanegara: funding acquisition, conceptualization, supervision, project administration.

Conflict of Interest

The authors declare no conflict of interest related to the manuscript.

Declaration of AI

The authors declare that no AI is used in the manuscript production.

References

Ali, L. (2019). Cyber crimes-a constant threat for the business sectors and its growth (a study of the online banking sectors in gcc). *Journal of Developing Areas*, *53*(1), 253–265. https://ideas.repec.org/a/jda/journl/vol.53year2019issue2pp.253-265.html

Attawna, M. D., Doan, T., Shadi, Doan, T., Pham, F., & Nguyen, G. (2023). Leveraging data plane acceleration for network coding deployment: A measurement study. *Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 34–39. https://doi.org/10.1109/NFV-SDNS9219.2023.10329588

Azari, A., Salehi, F., Papapetrou, P., & Cavdar, C. (2022). Energy and resource efficiency by user traffic prediction and classification in cellular networks. *IEEE Transactions on Green Communications and Networking*, *6*(2), 1082–1095. https://doi.org/10.1109/TGCN.2021.3126286

Belkhiri, A., Pepin, M., Bly, M., & Dagenais, M. (2022). Performance analysis of dpdk-based applications through tracing. *Journal of Parallel and Distributed Computing*, *173*. https://doi.org/10.1016/j.jpdc.2022.10.012

Çelebi, M., Özbilen, A., & Yavanoğlu, U. (2023). A comprehensive survey on deep packet inspection for advanced network traffic analysis: Issues and challenges. *Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi*, *12*(1), 1–29. https://doi.org/10.28948/ngumuh.1184020

Chairina, C., & Tjahjadi, B. (2023). Green governance and sustainability report quality: The moderating role of sustainability commitment in asean countries. *Economies*, *11*(1). https://doi.org/10.3390/economies11010027

Deri, L. (2021). *Using ndpi for monitoring and security* (tech. rep.) (Accessed: 2025-09-18). ntop.org. https://www.ntop.org/

Deri, L., Cardigliano, A., & Fusco, F. (2024). Advancements in traffic processing using programmable hardware flow offload. https://arxiv.org/abs/2407.16231

Domingos, J. M. F., Marques, D. G., Campos, V., & Nolasco, M. A. (2024). Analysis of the water indicators in the ui greenmetric applied to environmental performance in a university in brazil. *Sustainability*, *16*(20). https://doi.org/10.3390/su16209014

Feliana, F., Harwahyu, R., & Overbeek, M. V. (2023). Multichannel slotted aloha simulator design for massive machine-type communication (mmtc) on 5g network. *International Journal of Electrical, Computer, and Biomedical Engineering*, *1*(2), 72–102. https://doi.org/10.62146/ijecbe.v1i2.8

Ghosh, A., & Senthilrajan, A. (2019). Research on packet inspection techniques. *International Journal of Scientific Technology Research*, *8*, 2068–2073.

Hananto, A. L., Tirta, A., Herawan, S. G., Idris, M., Soudagar, M. E. M., Djamari, D. W., & Veza, I. (2024). Digital twin and 3d digital twin: Concepts, applications, and challenges in industry 4.0 for digital twin. *Computers*, *13*(4). https://doi.org/10.3390/computers13040100

Harwahyu, R., Ndolu, F. H. E., & Overbeek, M. V. (2024). Three layer hybrid learning to improve intrusion detection system performance. *International Journal of Electrical & Computer Engineering (IJECE)*, *14*(2), 1691–1699. https://doi.org/10.11591/ijece.v14i2.pp1691-1699

Ibrahim, A. (2024, January). *Integration of machine learning models in a microservices architecture* [Master of Science]. Polytechnic Institute of Bragança. https://share.google/QE9izRn5hndunaxXB

Imran, A., Wahid, M. S. N., Baso, F., & Fadil, A. (2024). Development monitoring information system based on website and whatsapp gateway at sd telkom makassar. *Journal of Embedded Systems, Security and Intelligent Systems*, *5*(3), 240–248. https://doi.org/10.59562/jessi.v5i3.5193

Jha, A., Singh, S. R., & Meenakshi. (2022). Human–machine convergence and disruption of socio-cognitive capabilities. *International Journal of Next-Generation Computing*, *13*(3), 754–762. https://doi.org/10.47164/ijngc.v13i3.893

Jia, M., Komeily, A., Wang, Y., & Srinivasan, R. (2019). Adopting internet of things for the development of smart buildings: A review of enabling technologies and applications. *Automation in Construction*, *101*, 111–126. https://doi.org/10.1016/j.autcon.2019.01.023

Joarder, Y. A., & Fung, C. (2024). Exploring quic security and privacy: A comprehensive survey on quic security and privacy vulnerabilities, threats, attacks, and future research directions. *IEEE Trans. on Netw. and Serv. Manag.*, *21*(6), 6953–6973. https://doi.org/10.1109/TNSM.2024.3457858

Jonnavithula, S., Jain, I., & Bharadia, D. (2024). Mimo-ric: Ran intelligent controller for mimo xapps. *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2315–2322. https://doi.org/10.1145/3636534.3701548

Kemp, S. (2024, January). Internet use in 2024 [Accessed 2025-09-17]. https://datareportal.com/reports/digital-2024-deep-dive-the-state-of-internet-adoption

Kusrini, E., Whulanza, Y., Ramakrishna, S., & Nurhayati, R. W. (2023). Advancing green growth through innovative engineering solutions. *International Journal of Technology*, *14*(7), 1402–1407. https://doi.org/10.14716/ijtech.v14i7.6869

Özbay, Z. N., & Dalkılıç, M. E. (2023). Ndpi derin paket inceleme aracı üzerinde bir çalışma. *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, *16*(2), 137–146. https://doi.org/10.54525/tbbmd.1253700

Park, S., Flaxman, A., & Schultz, M. (2021). Impact of data visualization on decision-making and its implications for public health practice: A systematic literature review. *Informatics for Health and Social Care*, *47*, 1–19. https://doi.org/10.1080/17538157.2021.1982949

Ramey, H. (2024, May). *Understanding the limits of deep packet inspection for network traffic classification* [Master of Science]. University of Texas at El Paso. https://scholarworks.utep.edu/open_etd/4136/

Raza, M., Kazmi, S., Ali, R., Naqvi, M. M. A., Fiaz, H., & Akram, A. (2024). High performance dpi engine design for network traffic classification, metadata extraction and data visualization. *Proceedings of the International Conference on Advancements in Computational Sciences (ICACS)*, 1–6. https://doi.org/10.1109/ICACS60934.2024.10473274

Rehman, S. U., Giordino, D., Zhang, Q., & Alam, G. M. (2023). Twin transitions industry 4.0: Unpacking the relationship between digital and green factors to determine green competitive advantage. *Technology in Society*, *73*, 102227. https://doi.org/10.1016/j.techsoc.2023.102227

Sapp, S., Dorius, S., Bertelson, K., & Harper, S. (2021). Public support for government use of network surveillance: An empirical assessment of public understanding of ethics in science administration. *Public Understanding of Science*, *31*, 096366252110495. https://doi.org/10.1177/09636625211049531

Sarasola, T. F. D. B., GarcÃa, A., & Ferrando, J. L. (2024). Iiot protocols for edge/fog and cloud computing in industrial ai: A high frequency perspective. *International Journal of Cloud Applications and Computing (IJCAC)*, *14*(1), 1–30. https://doi.org/None

Sarhan, S., Youness, H., Bahaa-Eldin, A., & Taha, A. (2024). Voip network forensics of instant messaging calls. *IEEE Access*, *12*, 9012–9024. https://doi.org/10.1109/ACCESS.2024.3352897

Shirota, Y., Presekal, A., & Sari, R. F. (2019). Visualization of time series data by statistical shape analysis on fertility rate and education in indonesia. *Journal of Advances in Information Technology*, *10*(2), 60–65. https://doi.org/10.12720/jait.10.2.60-65

Vailshery, L. S. (2024, September). Number of internet of things (iot) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2033 [Accessed 2025-09-17]. https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

Vorbrodt, M. (2023, February). *Analyzing the performance of linux networking approaches for packet processing: A comparative analysis of dpdk, io_uring and the standard linux network stack* [Master's thesis, Linköping University] [Retrieved from DiVA portal]. https://www.diva-portal.org/smash/get/diva2%5C%3A1789103/FULLTEXT01.pdf

Wang, S., Qian, L., Yi, C., Wu, F., Kou, Q., Li, M., Chen, X., & Lan, X. (2024). Sadma: Scalable asynchronous distributed multi-agent reinforcement learning training framework. *International Workshop on Engineering Multi-Agent Systems (EMAS)*, 64–81. https://doi.org/10.1007/978-3-031-71152-7_4

Whulanza, Y. (2023). Cohering existing technology with greener and modern innovation. *International Journal of Technology*, *14*(2), 232–235. https://doi.org/10.14716/ijtech.v14i2.6435

Yen, T. A. (2021). Flexips: A keep-tracking scalable network function design and implementation. *2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT)*, 607–613. https://doi.org/10.1109/CECIT53797.2021.00112

Zagloel, T. Y. M., Harwahyu, R., Maknun, I. J., Kusrini, E., & Whulanza, Y. (2023). Developing models and tools for exploring the synergies between energy transition and the digital economy. *International Journal of Technology*, *14*(8), —. https://doi.org/10.14716/ijtech.v14i8.6906

Zhu, J., He, S., He, P., Liu, J., & Lyu, M. R. (2023). Loghub: A large collection of system log datasets for ai-driven log analytics. *Proceedings of the 2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*, 355–366. https://doi.org/10.1109/ISSRE59848.2023.00071