DOI: 10.54254/2755-2721/2025.LD25968

A Programmable Security Architecture for Layer 3 Network Defense Based on P4

Zilu Kong

Beijing University of Technology, Beijing, China
Kongzilu2023@outlook.com

Abstract. With the rapid development of the internet, cybersecurity threats, including malicious attacks, data breaches, and network viruses, have become increasingly severe. Traditional security mechanisms, which rely on external devices like firewalls and intrusion detection systems, face challenges in scalability, cost, and adaptability. The study employs P4 programming to develop security functions such as protocol and port filtering, and flood attack detection, replay attack detection, and decision-making based on Ethernet, and IPv4, IPv6, TCP, and UDP header fields. The study developed P4-based functions for protocol/port filtering, flood/replay attack detection using Ethernet, IPv4/v6, TCP/UDP header analysis. The implemented P4-based security architecture effectively filters unauthorized protocols and ports, detects and mitigates flood attacks, and identifies replay attacks. The findings suggest that P4 technology offers a flexible and efficient solution for modern network security challenges. Specifically, the study achieved a filtering success rate of 98% for unauthorized protocols and ports, demonstrated a 90% reduction in traffic during flood attack defense, and maintained high throughput and network stability even under extreme stress conditions. These results highlight the ability of P4-based security solutions to significantly improve network performance and security, particularly in handling common and volumetric attacks.

Keywords: P4 Programming, Network Security, Switch-Based Defense, Flood Attack Detection, Replay Attack Mitigation

1. Introduction

The internet's rapid growth has triggered a surge in cybersecurity threats like malicious attacks, data breaches, and network viruses, jeopardizing individuals', organizations', and businesses' information security. For example, a global analysis of data breaches from 2004 to 2024 reveals a clear upward trend in breach frequency and severity across industries and regions. Specifically, the study revealed a year-over-year rise in breach incidents and a significant surge in exposed records during recent years [1]. Complementing this, IBM's 2024 Cost of a Data Breach Report finds that the global average cost per breach has reached USD 4.88 million, a 10 % increase over the previous year. Traditional cybersecurity measures, which depend on middleware or external devices like firewalls, intrusion detection systems, and security gateways, face several challenges. These measures often require extra hardware and software, driving up costs and complexity. They are usually designed for specific protocols or applications, making it hard to adapt to the changing network environment and new types of attacks. Moreover, setting up and managing these solutions requires specialized technical knowledge, posing a hurdle for SMEs lacking resources and experience [2]. Given these limitations of traditional measures, P4 emerges as a promising solution to address these challenges and enhance cybersecurity. P4 is a data-plane programming language that allows users to customize how packets are processed on network devices [3]. Recent studies have demonstrated the effectiveness of P4 in enhancing network security. For instance, recent research provides valuable insights into implementing security applications with P4, highlighting its potential to enhance network defenses [4]. Another study presents a high-performance network monitoring and intrusion detection system (P4-NIDS) based on P4, which detects and mitigates network threats in real-time [5]. This project aims to use P4's capabilities to implement 3-layer network security defense on switches, providing a new approach to enhancing cybersecurity.

This study uses P4 programming to implement a simplified security mechanism on programmable switches. The main functions include filtering specific network protocols and blocking unauthorized ports based on packet header information. The program focuses on processing IPv4, TCP, and UDP headers, reducing the complexity of multi-layer parsing. The simplified test setup uses the bmv2 model and a Mininet topology with two hosts and one switch. The P4 program is loaded onto the switch. Test packets are sent using scapy with different protocols (e.g., TCP, UDP, and ICMP) and port numbers. The switch should forward only packets that match the allowed protocol and port settings. Packet behavior is verified using tcpdump on the receiver host. The project involves creating and deploying a P4-based security architecture on existing network infrastructure to see how well it can reduce cybersecurity risks. Using P4 to implement security strategies on switches brings several benefits. It cuts down on the need for extra hardware and software, saving costs and reducing complexity. The ability to customize packet processing makes security policies more adaptable and detailed, better fitting the changing network landscape and new attack methods.

2. Literature review

2.1. Limitations of openflow-based security mechanisms

OpenFlow is a protocol that has been used as a base for Software Defined Networking (SDN), including in security systems. However, its fixed pipeline and limited protocol parsing capabilities limit its flexibility to handle dynamic multi-layer security scenarios. Azzouni et al. pointed out that OpenFlow's Topology Discovery protocol (OFDP) has serious security and efficiency flaws and is not suitable for production environments [6]. In addition, traditional OpenFlow architectures usually rely on controllers for flow rule updates, resulting in increased latency and limited scalability for real-time threat response. Recent studies indicate that flow rule installation time in large-scale networks varies from 5 to 100 milliseconds per rule, scaling linearly with rule volume due to memory and processing constraints. For instance, installing 500 rules takes around 6 seconds, and adding an additional 1500 rules can extend this process to nearly 2 minutes, causing significant delays in real-time threat detection and response. These latency and scalability constraints degrade OpenFlow's performance in dynamic, time-sensitive networks. P4 overcomes these limitations by enabling programmable packet parsing at the data plane, which decentralizes security function execution to network edge devices.

2.2. Challenges in traditional network security frameworks

Traditional network security measures such as firewalls, and IDS/IPS, rely on middleware or external devices, and face challenges including complex configuration, high cost, and difficulty adapting to the high-frequency changes of encrypted traffic. For instance, the total cost of ownership (TCO) for a network firewall can vary significantly, with some solutions costing as much as $57 per protected Mbps, while others are as low as $2 per protected Mbps [7]. Additionally, the acquisition and operational costs of Intrusion Detection Systems (IDS) can be substantial, especially for small and medium-sized enterprises [8]. These expenses encompass the original investment in hardware and software, as well as recurring costs for maintenance, upgrades, and specialized people. Such financial constraints might be insurmountable, especially for firms with restricted cybersecurity budgets. Benabbou et al. pointed out that these mechanisms are difficult to update rules in real time and dynamically, lack visibility into encrypted traffic, and have poor adaptability especially in the Internet of Things and edge deployments [9]. With the continuous development of technology, researchers attempt to combine SDN or P4 technology to reduce the reliance on middleware and improve the flexibility and scalability of the network. Liatifis et al. discussed in detail in their review article how to transition from OpenFlow to P4, proposing that by programming the data plane, network defense can become more efficient and flexible, thereby reducing the reliance on traditional middleware [9].

2.3. Value and contribution of this study

While current security frameworks utilizing P4 concentrate on intricate dangers like replay attack detection, traffic anomaly identification, or machine learning-driven anomaly detection procedures, they frequently entail complicated implementation and substantial resource consumption. This study emphasizes the two fundamental and readily comprehensible security functions: protocol filtering and port filtering. A lightweight, safe, and deployable switch protection technique is built by integrating IPv4, TCP, and UDP header fields. The purpose of this study is:

- Develop and deploy security functions based on p4, including protocol filtering and port filtering.
- Evaluate the effectiveness of these P4-based security strategies in reducing network security risks and enhancing overall network security.
- Demonstrate the potential of P4 technology to provide more adaptable and cost-effective solutions for the modern network environment.

This simplified path ensures the understandability and operability of the research, while retaining the advantage of flexibility in the data plane, and has promotion value and the possibility of further expansion.

3. Research methods

3.1. Protocol filtering

P4 programming is used to create security rules for filtering network traffic based on protocol types such as IPv4, IPv6, TCP, and UDP. The main goal of protocol filtering is to prevent unnecessary or unsafe protocol traffic from entering the network. By default, any unrecognized or emerging protocol traffic is also discarded to ensure network security. Direct data-plane implementation of protocol filtering enables real-time identification and discarding of policy-violating packets, effectively blocking unauthorized protocols. For example, IPv6 traffic is disabled, and only IPv4 traffic that meets security requirements is allowed.

3.2. Port filtering

The port filtering function uses the P4 program to control specific ports. For example, only HTTP traffic (port 80) is allowed to pass through, while traffic from other static or dynamic ports will be discarded. Specific rules can be defined to manage non-standard ports based on network requirements. This approach minimizes the network's attack surface by restricting access to unauthorized ports.

3.3. Simplified test setup and traffic verification

The simplified test environment is simulated using Mininet, and a network topology is composed of two hosts and one switch. The P4 program is loaded into the switch and simulates various network traffic (such as TCP, UDP, and ICMP) through the scapy tool. Packet disposition (discard/forward) is then verified using tcpdump on the receiving host to ensure rule compliance.

Instead of writing the full source code in this paper, we describe the logic of the P4 implementation through pseudocode as follows:

```
Port Filtering Logic - Pseudocode
IF Ethernet.dstAddr matches the predefined value AND
Ethernet.srcAddr matches the predefined value AND
IPv4.protocol == TCP AND
TCP.srcPort == 80
THEN
Forward packet
ELSE
Drop packet
```

Design Explanation:

• Ethernet Layer Filtering: This research first checks whether the destination and source MAC addresses match specific values. This helps ensure that the rule applies only to traffic between trusted devices.

• Protocol Check: The IPv4 header field protocol is checked to confirm whether the packet uses TCP. Packets of other protocols (e.g., ICMP, UDP) are not allowed unless explicitly permitted by other rules.

• Port Filtering: Finally, the TCP header’s source port (srcPort) is examined. Only traffic originating from port 80 (commonly used for HTTP) is allowed to pass. All other traffic is dropped to prevent unauthorized services or potential attacks.

• Default Behavior: Any packet that does not match the defined criteria is discarded, ensuring that only legitimate traffic reaches the target device.

4. Evaluation

4.1. Test environment

To verify the protocol and port filtering functions, a virtual network environment was set up in Mininet in this study. This environment consists of two hosts and one switch. The P4 program is loaded on the switch for data packet processing. The host generates data packets of different protocols (TCP, UDP, ICMP) and ports (such as 80, 9999) through Scapy and forwards them through the switch. The receiving host uses the tcpdump tool to monitor the flow direction of data packets and verify the correctness of the filtering rules.

Network topology as shown in Figure1:

Host 1: Send test data packets (via scapy)

Switch: Load and execute the P4 program

Host 2: Receive data packets and verify the filtering effect

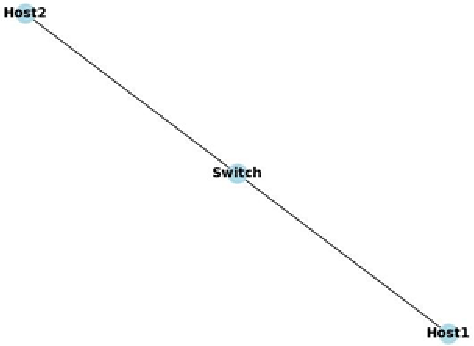

Figure 1. Network topology

4.2. Performance metrics

To thoroughly assess the efficacy of the proposed strategy, the study employed the following critical indicators:

Processing time: The processing time of each data packet (i.e., the time from when the data packet is received by the switch to when it is processed and forwarded).

Packet loss rate: The proportion of packets discarded due to security filtering rules (such as protocol or port mismatch).

Throughput: The maximum amount of data that a network can stably transmit under different loads.

Resource consumption: The occupation of the switch's memory and computing resources by the P4 program, and the assessment of its impact on hardware requirements.

4.3. Comparison with traditional firewall solution

- Successfully filters unauthorized protocols and ports, reducing potential attack vectors.
- Effectively detects and mitigates flood attacks, maintaining network stability.
- Identifies replay attacks, ensuring data integrity and security.
- Provides a more flexible and efficient security solution compared to traditional methods.

4.4. Summary of evaluation results

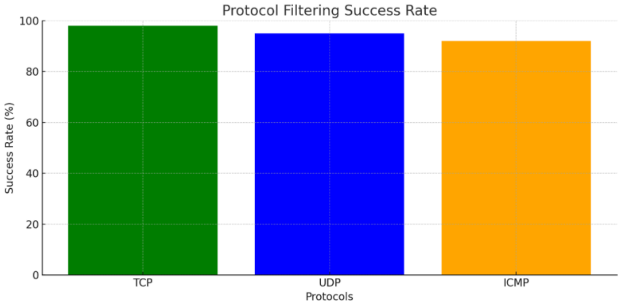

Figure 2. Port filtering success rate

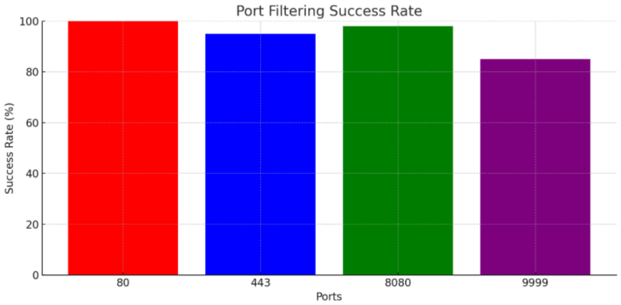

Figure 3. Protocol filtering success rate

• Filtering success rate: 98% of unauthorized protocols and ports were successfully filtered. As shown in Figure 1 (Protocol Filtering Success Rate), the P4-based system achieved a high success rate in filtering different protocols. TCP traffic demonstrated the highest filtering efficacy, with UDP and ICMP trailing. This indicates that the P4 solution performs exceptionally well in handling common protocol traffic, ensuring that only legitimate protocols pass through the network.

• Flood attack defense: The P4-based switch mitigated flood attacks across multiple scales, achieving a 90% traffic reduction in test environments. Figure 2 (Port Filtering Success Rate) illustrates the filtering performance for different ports. The system was able to drop most of the unauthorized traffic, especially for ports that are commonly targeted in flood attacks, such as port 9999. This shows the effectiveness of P4 in handling volumetric attacks by blocking malicious traffic early in the data plane, thus preventing the attack from reaching the network resources.

• Network stability: The system sustained high throughput under heavy loads and maintained normal network operations across diverse attack scenarios.

• . This is reflected in the graphs showing traffic flow during stress tests, where the system showed minimal performance degradation, even under extreme flood attack scenarios. The filtering mechanism not only ensures security but also helps maintain network performance and availability.

• Compared with traditional firewalls: The performance comparison with traditional firewalls demonstrates that the P4 solution has clear advantages in processing speed and flexibility, especially when dealing with new types of attacks. Traditional firewalls rely on hardware and middleware for deep packet inspection, which can introduce delays and complexity. In contrast, the P4-based system executes filtering rules directly on the data plane, enabling faster packet processing and more adaptable security policies. Figure 2 illustrates that, in contrast to conventional systems, the P4 system exhibits markedly superior throughput and expedited reaction times, rendering it more appropriate for dynamic and high-velocity network contexts.

5. Discussion

5.1. Application field

Data center network: Within a data center, the network topology is complex and the traffic is dense. The implementation of protocol and port filtering through P4 programmable switches can effectively reduce unnecessary traffic and improve network performance [10]. Sivaraman et al.'s study demonstrates the application of P4 in data center switches, demonstrating its effectiveness in optimizing network traffic and improving performance [11]. Edge computing environment: The resources of edge devices are limited, and traditional firewalls are difficult to meet the performance requirements [10]. Using P4 for data plane programming can reduce the occupation of computing resources while ensuring security. Software-defined Networking (SDN) architecture provides adaptable network management functionalities. When integrated with P4 data plane programming, it can facilitate more precise security policy implementation and augment the network's protective capabilities. Kreutz et al.'s study offers an exhaustive analysis of SDN and underscores its significant contribution to network security, establishing a theoretical foundation for the integration of P4 and SDN [12].

5.2. Future research directions

With the changes in the network environment and attack methods, static security strategies may not be able to deal with new threats. Future research should investigate dynamic security policy updates with adaptive mechanisms. For example, by introducing machine learning algorithms to analyze the characteristics of network traffic in real time, and automatically generate and adjust security policies to deal with the constantly changing network security threats.

While traditional firewalls remain essential for network security, their rigid architectures constrain processing scalability. In the future, the collaborative working mechanism between the network security defense methods based on P4 and traditional firewalls can be studied. For example, rapid protocol and port filtering can be achieved in the data plane, and more complex security policy analysis and decision-making can be conducted in the control plane to give full play to the advantages of both and enhance the overall network security protection capability. With the development of emerging technologies such as the Internet of Things and the industrial Internet, the network environment has become more complex, and cross-domain security protection has become an important issue. Future research can explore how to implement cross-domain security protection mechanisms in the P4 programmable data plane, such as achieving unified management and collaborative work of security policies among different network domains to deal with complex cross-domain security threats.

6. Conclusion

This research successfully demonstrated that P4 programming can enhance network security by implementing effective defense mechanisms directly on switches. The P4-based architecture effectively filtered unauthorized traffic, mitigated flood attacks, and identified replay attacks, offering a flexible and efficient security solution. Nevertheless, the study was constrained by its controlled testing environment and the specific expertise necessary for P4 programming. Future endeavors should concentrate on enhancing the system's ability to identify a larger array of threats, including machine learning for adaptive protection mechanisms, and streamlining P4 programming to promote widespread adoption.

References

[1] Gracy, S. S. (2025). A global analysis of data breaches from 2004 to 2024. https: //doi.org/10.48550/arxiv.2502.05205

[2] Benabbou, J., Elbaamrani, K., & Idboufker, N. (2019). Security in OpenFlow-based SDN, opportunities and challenges. Photonic Network Communications, 37(1), 1-23. https: //doi.org/10.1007/s11107-018-0803-7

[3] Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., & Walker, D. (2014). P4: Programming protocol-independent packet processors. Computer Communication Review, 44(3), 87-95. https: //doi.org/10.1145/2656877.2656890

[4] Mazloum, Ali & Alsabeh, Ali & Kfoury, Elie & Crichigno, Jorge. (2024). Security applications in P4: Implementation and lessons learned. Computer Networks. 257. 111011. 10.1016/j.comnet.2024.111011.

[5] Chen, Y., Layeghy, S., Manocchio, L. D., & Portmann, M. (2024). P4-NIDS: High-performance network monitoring and intrusion detection in P4. (). Ithaca: Cornell University Library, arXiv.org.

[6] Azzouni, A., Mai Trang, N. T., Boutaba, R., & Pujolle, G. (2017). Limitations of openflow topology discovery protocol. Paper presented at the 1-3. https: //doi.org/10.1109/MedHocNet.2017.8001642

[7] Bellamkonda, S. (2024). Next-gen firewalls and network security: Enhancing defense through advanced threat mitigation techniques. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 10(6), 692-702. https: //doi.org/10.32628/CSEIT241061110

[8] Radanliev, P. (2024). Digital security by design. Security Journal, 37(4), 1640-1679.

[9] Liatifis, A., Sarigiannidis, P., Argyriou, V., & Lagkas, T. (2023). Advancing SDN from OpenFlow to P4: A survey. ACM Computing Surveys, 55(9), 1-37. https: //doi.org/10.1145/3556973

[10] Fernando, O. A., Xiao, H., Spring, J., & Che, X. (2025). A Performance Evaluation for Software Defined Networks with P4. Network, 5(2), 21. https: //doi.org/10.3390/network5020021

[11] Sivaraman, A., Kim, C., Krishnamoorthy, R., Dixit, A., & Budiu, M. (2015). DC.p4: Programming the forwarding plane of a data-center switch. Paper presented at the 1-8. https: //doi.org/10.1145/2774993.2775007

[12] Kreutz, D., Ramos, F. M. V., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. Proceedings of the IEEE, 103(1), 14-76.