P4-Driven Network Security Improvements for Data Centers

Artun Horasan#1, Fatih Furkan Erkeç#2, Rida Doğrul#3, Erol Görkem Hanoğlu#4

, Didar Doğrul #5

*#Computer Engineering Department, Konya Food and Agriculture University
Konya/Meram*

1artun.horasan@ogr.gidatarim.edu.tr

2fatih.erkec@ogr.gidatarim.edu.tr

3rida.dogrul@ogr.gidatarim.edu.tr

4gorkem.hanoglu@ogr.gidatarim.edu.tr

5didar.dogrul@ogr.gidatarim.edu.tr

***Abstract*— More sophisticated network management and security solutions are required as modern data centers become more dynamic and complex. P4, a specialized language for programming network equipment, has demonstrated potential in overcoming these challenges. This paper investigates the ways in which P4 improves data center network security. We cover the fundamentals of P4, how it can be integrated with Software Defined Networking (SDN), and how it enables the development of adaptable and modifiable network solutions. We examine the structure of the P4 language, the compilation process, and how P4 programs are customized for different switches. We also review recent research on P4-based security solutions, with a focus on integrated defense mechanisms, availability, privacy, and encryption, and access control in particular. We also analyze the performance and behavioral characteristics of various P4 targets in data center environments, such as software switches, FPGAs, ASICs, and NPUs. We highlight new ideas and challenges of using P4 for data center security and provide perspectives on upcoming research paths. By grasping the capabilities and possibilities of P4, data center operators can use this technology to create more secure, efficient and adaptable network infrastructures.**

***Keywords*— P4, Data Center, Network Security, Software-Defined Networking (SDN), Programmable Data Planes**

1. Introduction

Data centers are expanding rapidly due to the technological developing on big data analytics, cloud computing, and artificial intelligence. This is placing a huge burden on network infrastructure. Fast networks that can expand and adapt to new security threats are essential for modern data centers. Conventional network devices frequently fail to meet these objectives due to their set characteristics and restricted flexibility.

Thanks to P4 (Programming Protocol-independent Packet Processors), this industry has undergone a transformation. It is possible to tailor network device packet processing with high-level programming languages like P4. It provides a level of control and adaptability that was previously unattainable with traditional, fixed-function hardware. Data center operators can utilize P4 to tailor the behavior of their network to meet specific security requirements, increase productivity, and react rapidly to new protocols and threats.

The capabilities of P4 are more and more enhanced through its integration with Software Defined Networking (SDN). SDN separates its planes by the control plane (responsible for managing the network and making decisions) from the data plane (responsible for forwarding packets). This division facilitates centralized management and control of the network. Then, simplifies the implementation and management of complex security measures. P4, being the programming language for the data plane, allows for the development of personalized packet processing pipelines that can implement these policies accurately and effectively.

This paper gives a detailed summary of P4's involvement in data center network security. We start by talking about the basics of P4, including its language structure, and how it is compiled for various switches. And we explore new studies on security solutions built on P4, examining topics like access control, privacy, encryption, availability, and integrated defense mechanisms. We also examine the differences in performance and behavior of different P4 targets within data center settings, such as software switches, NPUs, FPGAs, and ASICs. In conclusion, we emphasize the new ideas and difficulties involved in utilizing P4 for data center security and provide thoughts on potential future research areas. By the end of this paper, readers will have a solid understanding of how P4 is transforming data by the end of this paper, readers will have an idea about how P4 is transforming and making safer, more efficient and adaptable network structures on data center network security.

2. P4 USAGE IN DATA CENTER APPLICATIONS

Data centers, one of the cornerstones of modern IT infrastructure, are increasingly turning to programmable network technologies such as P4 to meet complex network needs. Programming Protocol-independent Packet Processors, a domain-specific language for network devices, allows free control over how data center devices (switches, NICs, routers, etc.) process packets, unlike the traditional model [1]. In this section, we will discuss how the P4 language and workflow can be used in data center environments.

2.1 P4 Language

P4 work teams [1] developed two versions, 14 [2] and 16 [3]. The first P4 language standard is the 14 standard. In version 14 of the P4 language, there are limitations such as inadequacy to describe and manage complex network protocols, usually due to the lack of sufficiently rigid structure of header structures and functional structures. In particular, these limitations experienced in the modularization capabilities of the language and the inadequacy of the portability and applicability of P4 programs running on different architectures of network devices cause sustainability and compatibility errors in large-scale and complex network applications. For these reasons, the 16 standard, which is a more advanced version compared to the 14 version, has been developed. Thus, it has a stricter language structure to meet the requirements of data center networks, and it supports many basic type operators as well as type operators that create derived types with a stronger type system and expression mechanisms. These; make the language more consistent, secure and flexible. In addition, since it is a static language version, declarative structures are more open and provide a better infrastructure to cope with the complexity of the language. While supporting different network architectures and targets, it supports multiple different pipeline architectures, allowing it to work on a wider range of hardware and increase the possibility of being applicable. The core of the P4 language; The rich structure it offers to define packet headers provides flexibility suitable for different network protocols and requirements. It provides mechanisms to define finite state machines and mapping tables used to parse and route network traffic, as well as defining the actions and sequences required to process network traffic, directing them to flow controllers to determine how these actions are implemented, and managing the overall operation of the network. For a detailed description of the P4 language standards 14 and 16, please refer to the P4 14 [2] and P4 16 language specifications [3].

2.2 P4 Workflow and Implementation

In data center applications, the use of the P4 language often provides network administrators with the ability to directly control and optimize network traffic. To implement a network application, users first program directly in the P4 language, and after compilation, the underlying equipment is configured to meet the user's functional requirements, completing the workflow.

In data center networks, P4 programs are usually loaded onto network equipment, and then compiled according to a specific target architecture. According to a specified P4 architecture, P4 programs can be deployed to achieve all the goals of the same P4 architecture, but cannot be transferred to different architectures. The core of the P4 language is; header definitions, parsers and tables, actions and flow controller components. When it comes to the compilation process, the P4 compiler converts P4 programs into binary configuration files in a format suitable for the target device by the target manufacturers to determine the functionality of the network equipment. Then, the P4 compiler converts the P4 program into a target-independent intermediate representation (IR) by representing the P4 program in a way that does not depend on the target hardware or software platform while preserving the general features of the language. In order to make the P4 program directly executable on a specific target device, it converts the target-independent intermediate representation into a language or code format that a specific target device can understand and provides mapping to the target.

P4 programmable structures, called P4 targets, represent the device on which the software or special hardware tools run on the CPU of the P4 programs are run.

A packet processing pipeline is defined specific to its target and architectural model. It also checks that the target device works correctly by generating the runtime mapping metadata necessary for the communication between the control plane and the data plane. After these steps, the optimization is completed. For example, there may be different compilation processes for solutions like PISCES [4], NetFPGA [5] or ASIC [7] based etc. Because each P4 compiler has a specified P4 target and is designed accordingly. Take the PISA architecture as an example to explain data processing. PISA (Protocol Independent Switching Architecture) is an architecture based on the P4 programming model and provides programmability of network switches and can be generally used to explain data processing. The input packet is processed with the help of many control blocks in the pipeline of the P4 program and is forwarded to the next control block in the pipeline and so on.

The PISA [8] architecture consists of three basic components: Parser, Match-Action Pipeline and Deparser. These basic components define the functionality of network switches and how P4 programs will work. In data center environments, such architectures are often used to meet the needs of high-performance packet processing.

The P4 runtime [9] is an Application Programming Interface (API) that provides the interaction between the control plane and the data plane of network devices. In data center environments, the P4 runtime is often integrated with network management software and automation tools. With this integration, devices based on different targets can be controlled with the same application programming interface. This makes the P4 runtime and application dynamic.

3. P4: NETWORK PROGRAMMING & PERFORMANCE

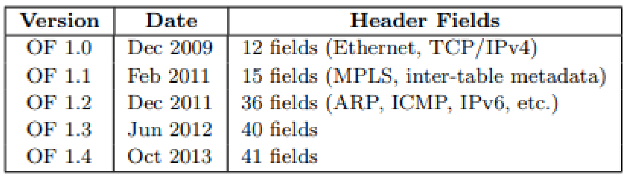
P4 is a high-level language designed for programming protocol-independent packet processors. It works alongside SDN (Software-Defined Networking) control protocols like OpenFlow. While OpenFlow offers some control over packet processing, it has limitations. The set of protocol headers it can handle has grown cumbersome, and it lacks the flexibility to accommodate new headers.

P4 addresses these shortcomings with three primary goals:

a. Reconfigurability in the field: Network operators should be able to update how switches process packets after deployment [10].

b. Protocol independence: Switches shouldn't be limited to specific network protocols [10].

c. Target independence: Programmers should describe packet processing functionality without needing to know the underlying hardware specifics [10].



Version	Date	Header Fields
OF 1.0	Dec 2009	12 fields (Ethernet, TCP/IPv4)
OF 1.1	Feb 2011	15 fields (MPLS, inter-table metadata)
OF 1.2	Dec 2011	36 fields (ARP, ICMP, IPv6, etc.)
OF 1.3	Jun 2012	40 fields
OF 1.4	Oct 2013	41 fields

TABLE 1. [1]

3.1 SDN and P4 working together

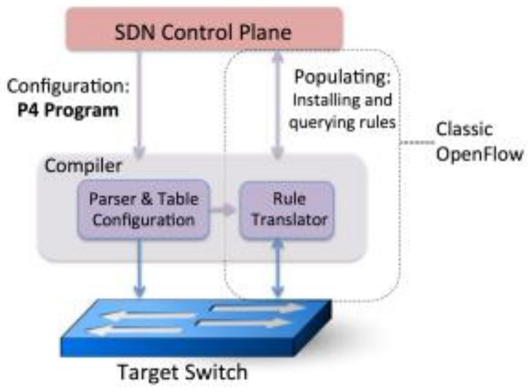
A single control plane may now manage numerous forwarding devices from different manufacturers thanks to SDN's separation of the control plane (network control) from the data plane (packet forwarding). P4's integration with SDN offers a more adaptive and flexible method of network administration.

- *Reconfigurability in the Field:* Following the first deployment, a programmer may alter the switch's packet processing path. Furthermore, the controller must have the ability to specify the parser packet and how the packet fields are processed [11].

- *Protocol independence:* a switch shouldn't be dependent on a particular packet or network protocol. The P4 programmer specifies the fields and headers that must be present in the program [11].

- *Target independence:* Regardless of the underlying hardware type, the programmer should be able to explain how well packets are processed. The compiler maps P4 onto an alternative platform, such as FPGA, ASIC, SOC, or GPU [11].

FIGURE 1. [10]



3.2 P4 program structure

A P4 program defines several key components:

- *Headers:* Define the structure and sequence of fields within a packet header, including field width and value constraints [12].
- *Parsers:* Specify how to identify headers and valid header sequences in packets [12].
- *Tables (Match+action):* The core of packet processing. These tables define conditions for

matching packets (based on header fields) and the actions to be taken when a match occurs [10].

- *Actions:* Basic building blocks for manipulating packets. These can be combined to create complex actions used within match+action tables [10].
- *Control Programs:* Determine the order in which packets are processed by different match+action tables. They are written as simple imperative programs using functions, conditionals, and table references [12].

By defining these components, P4 programs create a flexible and adaptable framework for managing packet processing in network devices [10].

3.3 Compiling P4 Programs for Diverse Network Devices

P4 programs offer a powerful and adaptable approach to managing packet processing in network devices. However, translating the programmer's intent into a format executable by specific hardware requires a compilation step. This section delves into the compilation process, highlighting how P4 programs are transformed for various switch architectures.

3.4 From Control Flow to Table Dependency Graphs

The P4 program's control flow section, written in an imperative language, conveniently outlines the switch's logical forwarding behavior. But it doesn't explicitly indicate dependencies between match+action tables or potential opportunities for parallel processing. To address this, the compiler steps in.

The compiler analyzes the control program to identify dependencies between tables and seeks opportunities to process header fields concurrently. This analysis results in an intermediate representation: a table dependency graph. This graph depicts the relationships between tables, exposing which tables can be executed independently and which require sequential execution due to data dependencies.

3.5 Target-Specific Backend Compilation

With the table dependency graph in hand, the compiler's target-specific backend takes over. This backend maps the abstract graph onto the specific resources available on the target switch. Here's where the versatility of P4 shines. P4 programs can be compiled for various switch architectures, including:

- *Software Switches:* Offering maximum flexibility, software switches allow complete control over table count, configuration, and parsing. The compiler directly maps the table dependency graph from the P4 program to the switch's tables. It leverages table type information to optimize table structures and matching criteria based on the switch's capabilities [13][14].

- *Hardware Switches with RAM and TCAM:* These switches utilize a combination of Random-Access Memory (RAM) and Ternary Content Addressable Memory (TCAM) for efficient packet processing. The compiler can configure hashing for RAM-based tables like the mTag table in edge switches, enabling fast exact matching. In contrast, core mTag forwarding tables that rely on matching a subset of tag bits might be mapped to TCAM for efficient lookups [13].

- *Switches Supporting Parallel Tables:* Some switches allow parallel processing of tables that don't have data dependencies. The compiler can identify such opportunities in the table dependency graph and arrange tables for parallel execution. For instance, in the mTag example, the mTag table and local switching table could potentially be processed simultaneously up to the point where the mTag is set [13][14].

- *Switches with Limited Action Points:* Certain switches restrict action processing to specific points in the pipeline, typically at the end. The compiler can address this limitation by instructing intermediate stages to generate metadata that is used to perform the final actions. In the mTag example, the compiler could indicate whether the mTag should be added or removed using metadata, allowing the final action to be performed at the designated point in the pipeline [13].

- *Switches with Limited Table Capacity:* Not all switches offer an abundance of physical tables. The compiler can address this by strategically combining multiple P4 tables into a smaller number of physical tables on the switch. In the mTag example, the local switching table could be merged with the mTag table. The compiler's rule translator would then combine rules from the two P4 tables during runtime to generate rules for the single physical table [13].

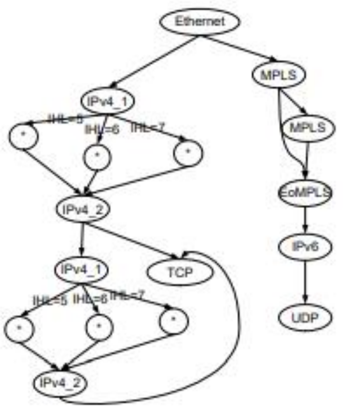


FIGURE 2. [4]

3.6 Packet Parsing: A Bottleneck in High-Speed Routers

Packet parsing is the process of extracting key fields from packets, such as IP destination address and TCP ports, which are used for packet forwarding decisions. While traditionally a straightforward task, the increasing complexity of packet headers due to the use of shim layers (e.g., MPLS, 802.1Q, MAC-in-MAC) has made parsing a bottleneck in high-speed routers. This article explores the challenges of packet parsing and introduces Kangaroo, a flexible packet parsing system designed to address these challenges [13].

3.6.1 The Rise of Parsing as a Bottleneck

In the past, packet parsing was a relatively simple process, involving the extraction of fields from a small number of fixed-length headers. However, the growing need for features like traffic engineering, security, and application-aware routing has led to the widespread use of shim layers. These layers add additional headers to packets, which can significantly increase the worst-case path length in the parse tree. This, in turn, increases the processing time required for parsing [10][13].

3.6.2 Challenges of Lookahead in Packet Parsing

One approach to improve parsing speed is to use lookahead, which involves processing multiple protocol headers in a single step. However, lookahead introduces two main challenges:

a. Dependencies: The decision of which headers to process in a single step depends on the values of fields in previous headers. For instance, the parser must determine whether the initial header is 802.1Q to parse an MPLS header that comes after an 802.1Q header. This could result in intricate decision-making [13][15].

b. Variable Length Headers: Variable-length headers are a feature of some protocols, including TCP. This makes it challenging to anticipate the offset of the subsequent header, which is essential for lookahead [15][13].

3.6.3 Kangaroo: A Flexible Packet Parsing System

Kangaroo is a packet parsing system designed to overcome the challenges of lookahead. It uses a Content Addressable Memory (CAM) to perform lookahead by simultaneously extracting and matching multiple fields from the packet. The CAM is configured with entries that represent specific sequences of header types. When a match is found, the parser knows exactly which fields to extract next.

To handle dependencies, Kangaroo uses a non-uniform lookahead approach. This means that the number of headers processed in a single step can vary depending on the path being traversed in the parse tree. Additionally, Kangaroo employs a dynamic programming algorithm to calculate the optimal amount of lookahead for each path.

For variable-length headers, Kangaroo avoids lookahead whenever possible. Instead, it uses the ALU (Arithmetic Logic Unit) to calculate the offset of the next header based on the extracted header length field. For short, variable-length headers, Kangaroo adds extra CAM entries for each possible header length.[13]

Current State	**Lookup Value**	**Next State**
vlan	0xaaaa	mTag
vlan	0x800	ipv4
vlan	*	stop
mTag	0x800	ipv4
mTag	*	stop

TABLE 2. [4]

3.6.2 Optimizing TCP Performance with Programmable Data Planes

The increasing requirements of contemporary internet applications call for a dependable data transfer technology. Many applications employ the Transmission Control Protocol (TCP) because to its adaptability to changing network conditions and resistance to many failure kinds. However, because they are closed systems with little control over specific network events, typical network switches hinder TCP performance.

The emergence of P4-programmable devices empowers developers to rapidly design and test customized solutions that enhance TCP performance. These solutions leverage functionalities like:

- *Fine-grained telemetry:* Programmable data planes enable in-band network telemetry, providing detailed data for troubleshooting and taking actions in response to network issues. Legacy monitoring solutions often miss crucial information due to their coarse-grained nature [13].

- *Traffic isolation:* P4 allows for traffic segregation based on policies like latency, packet drops, bandwidth allocation, and Quality of Service (QoS) requirements. This enables efficient resource utilization and avoids unfairness between competing TCP flows that use different congestion control algorithms [13].

- *Fast reaction to congestion:* Unlike legacy devices requiring manual configuration adjustments, programmable data planes allow developers to define specific actions to mitigate congestion, high latency, or poor link utilization. This can involve techniques like fast rerouting to backup paths when the primary link becomes congested [13].

- *Protocol and application offloading:* P4-programmable Network Interface Cards (NICs) can offload tasks like the TCP three-way handshake from server CPUs, improving overall application performance. Additionally, microservices-based cloud applications can potentially be entirely implemented within programmable NICs [13].

- *Microburst detection:* Short-lived traffic spikes, known as microbursts, can overwhelm switch buffers and lead to packet loss. Legacy devices lack the visibility to detect these events. P4 enables fine-grained measurements to monitor queues, detect microbursts, and take corrective actions to minimize packet loss and congestion [13] [17].

3.6.3 P4 and Network Programming Paradigms

The traditional network industry follows a bottom-up development process for protocols. P4, however, introduces a top-down approach, facilitating experimentation with novel ideas for programmable data planes. The P4 language provides a standardized way to describe packet processing independently of the target hardware.

3.6.4 Benefits of P4-based Solutions

- *Increased TCP performance:* P4-based solutions improve TCP performance measures including throughput, latency, and fairness by addressing the shortcomings of legacy devices**.**

- *Flexibility and quick development:* P4's programmable nature enables more quicker solution development, testing, and implementation for specialized network requirements.

- *Fine-grained visibility and control:* P4 gives more insight to network events and more control over network, allowing to make better decisions.

3.6.5 Challenges and Future Directions

While P4 offers significant advantages, there are challenges to address:

- *Limited device availability:* P4-programmable devices are still not as widely deployed as legacy devices [16][13].

- *Programming complexity:* Programming P4 requires specialized skills and expertise [16][12].

- *Standardization:* Ongoing efforts are needed to further standardize P4 and related tools for broader adoption [16][13].

P4 is a potent technology that is revolutionizing network programmability despite these difficulties. We should anticipate seeing even more creative solutions that make use of P4 to maximize TCP performance and provide an enhanced networking experience as the technology develops and becomes more extensively used.

3.7 Enhancing Transmission Capacity Using Programmable Data Planes

High performance and dependable data transport are critical in the world of internet applications. The Transmission Control Protocol (TCP) intervenes in this situation to guarantee data integrity and adjust to network circumstances. However, traditional network switches limit TCP's potential due to their inflexibility.

Enter P4-programmable data planes, empowering developers to design and deploy customized solutions that enhance TCP performance. These solutions leverage functionalities like:

- *Fine-grained telemetry:* P4 enables in-depth network monitoring, providing valuable data for troubleshooting and taking real-time actions to address network issues [16].

- *Traffic isolation:* P4 allows for segregating traffic based on specific criteria, like latency, packet drops, bandwidth allocation, and Quality of Service (QoS) requirements. This fosters efficient resource utilization and avoids unfairness between competing TCP flows [16]

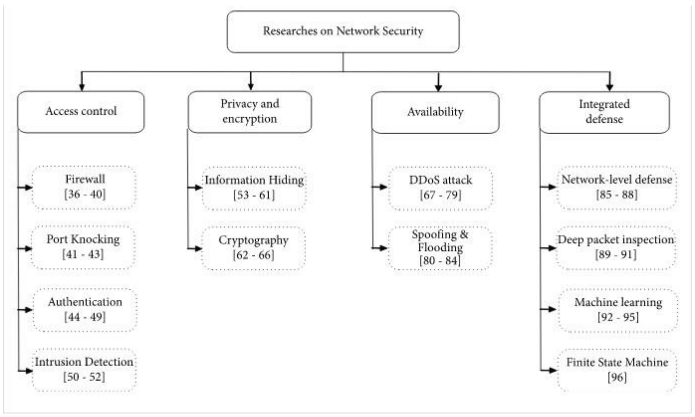
- *Fast response to congestion:* Unlike traditional devices requiring manual adjustments, P4 grants developers the ability to define actions to mitigate congestion, high latency, or poor link utilization. This can involve techniques like rerouting traffic to backup paths when congestion arises [16].

- *Protocol and application offloading:* By handling tasks like the TCP three-way handshake, P4-programmable Network Interface Cards (NICs) can increase server CPU utilization and enhance application performance [16].
- *Microburst detection:* Microbursts are brief bursts of traffic that overflow switch buffers, leading to packet loss [17] [16].

4. P4 BASED NETWORK SECURITY

Several advancements in network security may be achieved with the processing power of P4 programmable data centers packet-based inspection and filtering technologies. Thus, P4 switches can execute several tasks including regulating incoming network traffic, preventing intrusions, encrypting data, and utilizing multi-layered defense technologies to guarantee network security. A thorough map of P4 programmable data center-based network security research is shown in the image.

Figure: P4 programmable data center-based network security research map.


Researches on Network Security
Access control
Privacy and encryption
Availability
Integrated defense
Firewall [36 - 40]
Port Knocking [41 - 43]
Authentication [44 - 49]
Intrusion Detection [50 - 52]
Information Hiding [53 - 61]
Cryptography [62 - 66]
DDoS attack [67 - 79]
Spoofing & Flooding [80 - 84]
Network-level defense [85 - 88]
Deep packet inspection [89 - 91]
Machine learning [92 - 95]
Finite State Machine [96]

Access control, privacy and encryption, availability, and integrated defense are the four primary categories under which P4-based network security applications are categorized in this context. While firewalls, port knocking, and authentication are all part of access control, information concealing and cryptography solutions come under the category of privacy and encryption. The availability section describes flood attack prevention strategies and DDoS assault defenses. Techniques like machine learning, deep packet inspection, network-level defense, and finite state machines are examples of integrated defense.

4.1. Access Control

Limiting access to items or digital resources is known as access control. Owing to the benefits that P4 programmable keys provide for data centers, the data center can handle tasks like authorization and authentication. The server is relieved of the load of processing copious quantities of intrusion information by pre-filtering untrusted traffic. Consequently, business applications may be processed more quickly by utilizing the server's computational resources. Access control is essential for enhancing security and more efficiently managing network traffic in P4-based data centers.

For instance, because of P4's speed and versatility, research on the management and filtering of incoming traffic makes it simpler to identify and reject suspicious packets. By doing this, data centers' security risks are reduced and performance degradation is avoided. Furthermore, the P4 programming language allows access control lists (ACLs) to be continuously updated, allowing for continuous optimization of network traffic control.

4.1.1. Firewall

In data centers, P4-based programmable switches are crucial to maintaining network security. Incoming traffic is efficiently managed and filtered by these switches. First, an Ethernet, IPv4/IPv6, UDP, and TCP header parser was employed in a firewall solution in P4 [50]. There are two MATs (Matching-Action Tables): one uses blacklist filtering (barred list) and the other uses whitelist filtering. Direct packet drops occur when a packet matches the blacklisted network host's MAC/IP address.

Examples:

CoFilter: CoFilter[51] uses precise matching table entries to determine the stream descriptor by directly applying a hash function to each packet's 5-tuple (array of five). Streams that conflict are mapped to the new directory. Hash collision computations are carried out on the server, which needs a lot of memory and computing power. A copy of the packet at the start or end of the flow is transmitted to the server for further processing while other packets are handled on the switch pipe.

VNGuard and P4Guard: Features like virtual machine migration, virtual network topology design, and customized security needs are not supported by traditional firewalls. Consequently, VNGUARD, a software-based virtual firewall, was proposed by Deng et al. [52]. The SDN and NFV-based policy language defined by VNGuard is capable of managing virtual firewall settings in accordance with virtual network needs. P4Guard [53] is a virtual firewall based on P4 that is adjustable, taking the role of the software-based firewall in the VNGuard system. The controller of P4GUARD gathers traffic data from switches and keeps an eye on network traffic. Firewall rules dictate how packets are processed in the data center.

5G Multi-Tenant Firewall: A fully functional 5G multi-user firewall based on FPGA is proposed by Ricart-Sanchez et al. [54] to manage GTP (GPRS tunneling protocol) data sent between the edge and the core network. The TCAM table allows the firewall to distinguish between various tenants and end users by parsing and matching GTP header information using the P4 programmable data center. This is done by adding two parameters to the regular five parameters. The end user ID is indicated by the GTP ID, but the tenant ID is given by the VNI (VXLAN Network Identifier) in the VXLAN header.

Table 3: P4 Based Firewall Applications in Data Centers

Feature	Application	Explanation
Basic Protocol Support	Ethernet, IPv4/IPv6, UDP, TCP, Firewall	Recognizes and can handle basic network protocols.
Filtering Method	Whitelist, Blacklist	Separates safe and unsafe traffic.
Stream Descriptor	CoFilter	Calculates stream identifiers using a hash function.
Virtual Firewall	VNGuard, P4Guard	Provides SDN and NFV based virtual security solutions.
5G Support	FPGA Based Solution	Provides multi-tenant security in 5G networks.

The benefits and applications of P4-based firewalls in data centers are demonstrated by these examples and solutions. These firewalls' excellent performance and versatility greatly boost data center network security.

4.1.2. Port Knocking

A quick and clever way to open network ports for authentication is by port knocking. Because it may be implemented locally and does not require a controller to safeguard all network data, this technique is ideal for data centers. A plan is put out by Almaini et al. [55] to assign this task to the data center. Connection permissions are only given to sites that are able to communicate the right port sequence. These permissions expire after a set amount of time, at which point the client needs to re-authenticate.

For port knocking on the P4 switch, a finite state machine (FSM) with a finite number of states is utilized. The FSM can change states once it receives the proper port number; if it receives the wrong sequence number, it goes back to the first state. During P4 functioning, metadata records status data. The safe distribution of the shutdown command to reliable nodes and the ease with which packet snooping can compromise unencrypted port deactivation are left unsaid.

Based on the varying levels of port theft function offloading, P4Knocking [56] suggests four implementation strategies. They are as follows:

:

1. *Full Data Plane Offload:* This technique operates at line speed for all operations.
2. *Hybrid Data and Control Plane Offload:* This group includes the third and second techniques.
3. *Controller-Only Implementation:* Using this technique, the disablement order's table rules are the only ones loaded by the control plane.

Maybe the best course of action is to empty the data center totally in terms of performance. Port theft activities are always carried out at line speed. The master port disable verification is not loaded by the control plane; it just loads table rules for the disable order. Consequently, the data center can confirm the hit order even in the event that the link to the control plane is lost.

To guarantee that only authorized devices can access the server in data centers, two authentication schemes—port stealing and port scanning throttling—are suggested, both of which leverage P4. According to experimental findings, these techniques have no appreciable detrimental effects on switch performance and increase network availability overall by relieving the controller of some of its responsibilities and lowering communication overhead between the control plane and data center.

Table 4: P4 Based Port Knocking Applications in Data Centers

Feature	Application	Explanation
Authentication	FSM Usage	When the correct port order is received, it changes state, when the wrong order is received, it returns to the initial state.
Unloading	Full Data Plane, Hybrid Data and Control Plane, Controller	Offers different degrees of unloading.
Security	Lack of Encryption	Unencrypted port stealing can be affected by packet sniffing.
Performance	Line Speed Processing	All operations are performed at line speed, providing high performance.

These techniques show the benefits of various ways to improve network security in data centers using P4-based programmable switches. Data centers rely heavily on port knocking as a safe and reliable authentication technique.

4.1.3. Authentication

A key component of maintaining security in data centers is authentication. In data centers, there are several BYOD possibilities, much like with workplace and campus networks. In these situations, P4-based systems increase data center security effectively by giving the data center control over authentication procedures.

Poise: Poise [58] offers context-aware techniques to offer protection mechanisms in Bring Your Own Device (BYOD) scenarios, based on P4 devices. Contextual information is added to the packet header by the BYOD client. When a device runs the Poise software, it parses the data in the packet header and applies an ACL (Access Control List) according to the runtime context of the device. Poise has a modest overhead compared to OpenFlow-based protection systems, and it can respond nimbly to control plane saturation assaults. On the other hand, the system can be susceptible to replacement packet and manipulation attacks because the context packet isn't controlled.

P40f: Passive fingerprint solutions based on software are now unavailable and unable to handle the volume of data on high-speed networks. P40f, a fingerprint recognition system built on a programmable data center that can execute accept, release, and forward security rules at line speed, is proposed by Bai et al. [59]. P40f gathers and maintains metadata pertaining to the parser's corresponding action pipelines and the p0f signature field. The lookup table then uses the information as keys. When a rule matches, the packet is handled according to the matching rule's instructions. The packet is then sent to the next destination by the P40f once it has parsed the packet header. P40f is devoid of an intricate plan that restricts the pace of traffic.

E-Replacement: The detection performance of existing SDN port scanning algorithms suffers from a lack of consideration for hash conflicts. E-Replacement, a novel browser data gathering technique suggested by Cai et al. [60], can lessen the performance deterioration brought on by hash dispute. To lessen storage burden, the E-Replacement scheme's key logs the features of the scanning activity and retains only data that raises red flags. In order to do away with the need for data tagging, the controller regularly obtains statistical data from the switches and classifies the data using an unsupervised machine learning method.

SYN Proxy: To stop SYN Flood DDoS assaults, Scholz et al. [61] provide a SYN proxy method based on SYN cookie or SYN authentication from P4. SYN authentication only updates the relevant fields during the handshake procedure, making it relatively easier to implement than the SYN cookie scheme, which modifies the relevant fields of each packet in a totally transparent manner to the TCP client. Based on the protocol flag, the broker-reserved state, and the used approach, the SYN proxy alters the received packet. Changes to the serial number and status whitelist are organized into a corresponding action table.

P4DAD: Six recognized anti-spoofing solutions [62] have been put into practice to prevent IP address spoofing, and they are based on the NetFPGA SUME P4 platform. The resource consumption of several anti-spoofing techniques on the FPGA was compared by the author. Throughput of around 8.5 Gbit/s and processing delay of about 2 μs per packet were attained by these approaches.

P4DAD: As a straightforward substitute for encryption or authentication, P4-based Dual Address Detection (P4DAD) [63] offers a way to filter bogus NDP (Neighbor Discovery Protocol) signals. Binding entries between IPv6 addresses, port numbers, and address states are kept up to date via P4 switch registers. Generally speaking, message spoofing attack packets do not adhere to the aforementioned binding connection. Consequently, P4DAD does not need to change the host protocol or NDP in order to identify and remove malicious NDP messages.

Table 5: P4 Based Authentication Applications in Data Centers

Feature	Application	Explanation
Authentication	Poise, P40f	Context-aware strategies and fingerprint recognition systems in BYOD scenarios.
Data collecting	E-Replacement	It records the characteristics of browsing behavior and keeps suspicious data.
DDoS Protection	SYN Proxy	Provides protection against DDoS attacks with SYN cookies and SYN authentication.
Fraud Protection	P4DAD	Provides authentication by filtering out forged NDP messages.

These techniques demonstrate the benefits and potential uses of P4-based authentication systems in data centers. These technologies play a significant role in data centers as a reliable and safe means of authentication.

.

4.1.4. Intrusion Detection

In data centers, intrusion detection is essential to preserving network security. Network traffic may be effectively monitored and abnormalities can be detected using P4 programmable switches. Data center-specific intrusion detection systems (IDS) accomplish this goal by ensuring security and enhancing efficiency.

Two-Level IDS: Ndonda et al. [64] suggest a two-level Intrusion Detection System (IDS) for networks connected to Industrial Control Systems (ICS). Packets are parsed and handled by P4 switches in accordance with a first-level whitelist. Suspicious packets are sent to second-level devices for further examination rather than being discarded outright. In order to inform the controller to either accept or reject the connection, a chosen host assesses if the packet is malicious at the second level and modifies the switch's filter.

P4ID (P4 Enhanced Intrusion Detection): Ndonda et al. [64] suggest a two-level Intrusion Detection System (IDS) for networks connected to Industrial Control Systems (ICS). Packets are parsed and handled by P4 switches in accordance with a first-level whitelist. Suspicious packets are sent to second-level devices for further examination rather than being discarded outright. In order to inform the controller to either accept or reject the connection, a chosen host assesses if the packet is malicious at the second level and modifies the switch's filter.

Statistical Analysis for Anomaly Detection: A method for statistically analyzing packet execution pathways in the data center is put forth by Sanghi et al. [66]. A packet in a P4 schema can follow any path (implemented tables, performed operations). Paths may be traced and anticipated and observed distributions can be produced using the Ball-Larus coding approach. The switch aggregates the packet execution path distribution of real traffic over time and maintains per-path statistics using a hash table. To identify aberrant modes, the traffic distribution is compared to a predicted distribution.

Table 6: P4 Based Intrusion Detection Applications in Data Centers

Feature	Application	Explanation
Multi-Layered Approach	Two Level IDS	Two-level security with P4 switches and dedicated hosts.
Pre-Filtering	P4ID	Pre-filtering in the data center to reduce IDS load.
Anomaly Detection	Statistical analysis	Statistical analysis on package execution paths.

These techniques demonstrate the benefits of and uses for P4-based intrusion detection systems in data centers. These products play a significant role in data centers as a safe and efficient intrusion detection technique.

4.2. Privacy and Encryption

Anonymity and user privacy are crucial in data center networks. A major security risk might arise if the packet content contains sensitive information about the user's identity and online transactions. Two important techniques for limiting and protecting access to information are encryption and confidentiality. These strategies frequently entail data transformation, encryption, or encoding. The encryption and privacy systems in use today, however, have a number of significant limitations. First of all, altering the structure of the internet as a whole to incorporate certain encryption and privacy features is almost impossible.
Additionally, because the majority of solutions are software-based, controlling high-speed data flow can be challenging. As a result, the usefulness and scope of encryption and privacy strategies may be restricted. Recently, research attempts to develop encryption and lightweight anonymity systems have focused on programmable keys. This tactic can assist in removing current obstacles and offer more flexible and scalable alternatives. Nonetheless, study in this area is still needed as it is still evolving.

The most important studies and conclusions in the area of encryption and privacy are compiled in Table 2 [49]. These papers give light on future research directions and offer a useful resource for guaranteeing user privacy and security in data center networks.

4.2.1. Hiding information

Information concealing is crucial for protecting network traffic and thwarting assaults in data center networks. The information masking strategies applied in P4-based data center networks will be the main topic of this section.

By encrypting the network topology, NetHide [67] aims to lessen topology-aware assaults. This method applies heuristics and an integer linear programming (ILP) solver to the network complexity issue, treating it as a multi-objective optimization problem.

By masking the IP address, Protection Against Surveillance of Network Elements (SPINE) [68] eliminates eavesdropping from intermediary networks. Hosts and switches do not need to speak to one another since SPINE is data plane-based and not software-based. The IP address is encrypted by the sender, Prior to delivering a packet to an untrusted entity, the TCP sequence number and acknowledgment number are required.

Disconnection results from the IP address encryption technique used by Chang et al. [69]. By utilizing a dynamic key update strategy to shield the sender's true IP address from untrusted parties, this approach improves security.

A plan for updating security rules in accordance with various network traffic types is put out by Qin et al. [23]. Every time a fresh flow's initial packet reaches switch P4, the controller determines whether to activate the security policy and applies the required modifications.

MIMIQ, an IP address shuffling technique that may randomize IP addresses, is proposed by Govil et al. [71]. This solution consists of switches that update entries in routing tables and an address distribution server that disperses IP addresses.

LANIM, a learning-based network security method, is proposed by Liu et al. [72]. This system uses encryption techniques to stop eavesdropping and detects the network's condition.

Three-line protection and multipath packet transmission are features of P4NIS [73]. This method makes it harder to eavesdrop while also increasing transmission efficiency.

The adaptive multipath scheduling (AMS) technique is put out by Zhou et al. [74]. This method improves transmission efficiency by choosing alternative network pathways and making better use of available capacity.

PANEL (Practical Anonymity at Network Level) is a low-cost network anonymity solution proposed by Moghaddam et al. [75]. PANEL necessitates P4-based devices and provides several ways to conceal packet data.

These solutions offer diverse methods to strengthen network security and handle distinct facets of information hiding in P4-based data center networks.

4.2.2 Cryptography

Contemporary commercial programmable keys lack a unique cryptographic processor and only provide simple arithmetic and search functions. As a result, using Advanced Encryption Standard (AES) is challenging. In order to decrease the number of processes needed for AES encryption and implement AES on a Barefoot Tofino-based programmable key, Chen et al. [76] suggested a hash lookup table approach. The authors found that two rounds of AES may be completed by a single pipeline. In order to handle this, the system injects packets back into the pipeline using a packet fallback approach. The AES-128 algorithm may be finished in five pipeline passes in this manner.

IPsec is implemented using P4 keys in P4-IPsec [77]. In contrast to regular IPsec, a P4-IPsec tunnel is created and maintained by a controller using a tunnel configuration file that has already been specified. The Security Policy Database (SPD) and Security Alliance Database (SAD) are replaced in the P4 switch by the mapping transaction table. P4 extern uses an encryption suite that adds delay and decreases efficiency since it requires the key CPU to do encryption and decryption computations. Furthermore, further Linux client utilities need to be installed on the host since P4-IPsec can only be used in tunnel mode.

MACsec is implemented using P4-MACsec [78], which uses the P4 key. For layer 2 protection, MACsec employs symmetric encryption or cryptographic integrity checking. Its implementation necessitates an understanding of network topology and involves a laborious and intricate initial setup procedure. However, conventional network switches' support for the Link Layer Discovery Protocol (LLDP) finds it challenging to quickly identify topology changes. P4-MACsec improves LLDP for connection discovery and monitoring by encrypting payloads and sequence numbers. Encryption and decryption processes are kept as P4 externs in the P4 pipeline, whereas MACsec is done directly in the P4 data plane.

An online network traffic anonymization system (ONTAS) is proposed by Kim et al. [79]; it converts network operators' anonymization strategies into a unique method based on P4. This approach anonymizes the incoming packet stream at the line rate and synthesizes the matching process table in the data plane. The PISA-configured switch is used to construct ONTAS, which is scalable and adaptable and offers a policy language for expressing anonymous tasks.

A packet authentication method based on programmable data plane is proposed by Zuo et al. [80]. The Options field of the IP header contains a unique key that the sender appends to the packet. SDN based on OpenFlow gains a P4 switch. P4 switches decode the packet key identifier to enable proportionate sampling, which gives them complete control over packet delivery. The controller in OpenFlow switches checks the integrity of the sample packet and establishes flow rules for unusual packets. The matching space may be tailored and the transmission time is 2.5% lower with the suggested technique than with the comparison system.
Nevertheless, it is limited to identifying spoofing and altered communications; it is unable to pinpoint the precise location of the device used for spoofing or tampering. Security tags need to be added to the host along with a P4 key for instantiation.

4.3. Availability

Attackers flood the victim's bandwidth or computer resources with packets in an attempt to deplete them. DDoS assaults often take the shape of flooding or spoofing, and they can be of the following types:

1. *Slow DDoS:* Attackers send packets with similar power to regular traffic while still utilizing genuine TCP behavior, making it hard to identify the victim with conventional techniques.

2. *Volumetric attacks***:** These attacks mostly use UDP or the Internet Control Message Protocol (ICMP) to flood targets with attack traffic and fill available bandwidth.

3. *Malicious hosts:* They might incorporate an amplification technique into their assaults and issue requests that result in return traffic that far surpasses the sent traffic, perhaps generating a DDoS attack that surpasses Tbps, often known as a Reflection Amplification Amplification (RA) attack.

4. *In addition:* Since a genuine session is not necessary to carry out these types of attacks, attackers frequently use this spoofing approach to spoof the source IP address in reflection-based assaults in order to evade detection.

The strategy for fighting against attacks may be broadly broken down into three steps: detection, classification, and mitigation, despite the fact that assaults can take many different forms. DDoS defensive options are compiled in Table [49], with an emphasis on attack detection, attack scenarios, and attack mitigation.

4.3.1. DDoS Attack

The collaborative attack prevention system known as LAMP (Layer 7 Attack Prevention with Programmable Data Planes) is proposed by Grigoryan et al. [81]. Information from the application layer is necessary for this technique to work. Therefore, an attack flag is added to the IP option header field of a packet and the packet is delivered to the ingress switch if a host detects a DDoS assault at the application layer. All other packets in the flow are dropped by the entry switch once it gets the packet with the attack flag.

When there is a Telephone Denial of Service (TDoS) incident, the network is inundated with malicious SIP INVITE packets that prohibit customers from utilizing phone services. A preventative method against SIP proxy DDoS assaults termed TDoSD@DP is proposed by Febro et al. [82]. The quantity of INVITE and BYE messages is kept track of in the stateful P4 register. The SIP INVITE packet is destroyed if, after receiving a significant quantity of INVITE data, no BYE message is received. The detection and mitigation method of TDoSD@DP is straightforward, and it mitigates the attack nearest to the source and stops additional resource consumption brought on by attack packets entering the network.

The authors of the aforementioned study [83] provide a way to put into practice a distributed resource-based DDoS security system in order to identify assaults near their source. The packet count is recorded on each port of the P4 switch in this system. Every port's counted values are compared by the controller to a threshold. The P4 switch implements a rule to reject packets from the same port if the threshold is exceeded. Through the early detection of malicious traffic, this approach conserves bandwidth and processing power.

An FPGA accelerator device is presented by Kuka et al. [84] as a defense against reflection amplification attacks. It speeds up and improves the effectiveness of malicious traffic filtering in the backbone infrastructure before the attack traffic reaches the target. A P4 application that runs on the FPGA was implemented in VHDL and ported by the authors. After extracting the packet from the impacted segment of the incoming data, the device forwards it to a summarizer controller. The influence that each source IP address has on the attack is used by the heuristic method to identify the IP addresses of the attackers. The packet discard rule is loaded in the event that an attack is identified.resources.

Pushback [85] is a security technique that may recognize the pattern of attack traffic and restrict traffic near the assault source. However, Pushback is unable to identify more sophisticated and intelligent DDoS assault patterns because of the restricted processing power of conventional keys. Mi et al.'s [86] enhanced DDoS attack mitigation system, ML-Pushback (machine learning-based Pushback), makes use of machine learning technology. The P4 switch gathers the rejected packets in ML-Pushback and forwards them to a summator control plane. The control plane's deep learning module gathers summaries, extracts signatures from them, and uses a decision tree model to categorize the collected summaries. It limits attack traffic if it has been determined that there is attack traffic.

DDoS assaults often skew the distribution of IP addresses, however BUNGEE [87] uses entropy analysis of incoming packet addresses to identify DDoS attacks. A switch notifies upstream switches if it notices any entropy corruption. The devices upstream are responsible for filtering packets from the alert-raising device. In the end, these devices counteract the attack's consequences by acting as a detecting mechanism and warning upstream devices. To stop attack flows as near to the source as feasible, the aforementioned procedure is repeated.

On a programmable switch, Lapolli et al. [88] build an entropy-based DDoS detection system. The count-min charts data structure is used to estimate the frequency of source IP address and destination IP address in the observation window in order to fulfill the processing time requirement with limited store space. To estimate the entropy value and perform the computationally demanding arithmetic function, a memory-optimized longest prefix mapping (LPM) lookup table is employed. An attack warning is generated when the entropy value rises over a predetermined threshold. The aforementioned technique is put into practice on a software switch by the authors, who then evaluate it using an anonymous network monitoring dataset.

A distributed network defense architecture (DIDA) is proposed by Khooi et al. [89]. They demonstrate how each user's connection may be automatically monitored without using up the energy of the network controller by utilizing an effective data structure and a programmable stateful data plane. DIDA monitors each user's connection using the count-min charts data structure and a predetermined monitoring interval. It then analyzes the quantity of requests and answers to ascertain whether a device is delivering unsolicited attack packets to ISP network victims. The edge router use ACL to stop the attacker's incoming traffic when it detects a DDoS attempt.

A multifunctional DDoS protection method is implemented by Dimolianis et al. [44] by focusing on several aspects of TCP/UDP traffic. Three DDoS attack traffic characteristics are identified by this scheme: (i) the quantity of packets received in a specific time frame; (ii) the share of traffic received by a specific subnet overall; and (iii) the share of flow from the inlet to the output. Two sorts of instances are evaluated: cases with two features (F2), which contain just the first two characteristics, and cases with three features (F3). The results indicate that the first two elements are adequate to characterize DDoS in small-scale attacks, whereas additional features are preferable in the other two instances.

A DDoS detection and mitigation approach is proposed by Friday et al. [91] as a defense against sluggish and large-scale DDoS attacks. There are two components to the strategy. First, one-way traffic analysis is performed using bloom filters and time-dependent data. Then, a record of the bandwidth utilized by different apps is made. Data from statistics are used to define the dynamic threshold. A certain type of traffic is deemed harmful when its behavior above a certain threshold, in which case it has to be blocked or have its speed restricted. Using the API, administrators may put network characteristics into the data plane for use in calculating dynamic thresholds.

DoS attack packets often have distinct combinations of source MAC addresses and IP addresses because they originate from different source IP addresses or MAC addresses. DroPPPP [92] via P4 switch was proposed to minimize DoS assaults based on this observation. Using this approach, the recorded hash value is compared to the hash of the source IP address and MAC address. An assault is identified if there isn't a match and the last attack occurred during the past five seconds. The precomputed and forecasted values are stored in the data plane key table using TCAM.

INDDoS is a large-scale in-network DDoS victim identification technique presented by Ding et al. [93]. This method uses the BACON table to record the total number of target IP addresses. A device is labeled as a victim if its IP address is impacted by several source IPs in the data plane that surpass a threshold value in a predetermined amount of time. Due to physical resource constraints, Ding et al. [93] outline the adjustments needed to implement INDDoS on a Tofino switch and modify the settings to ideal levels derived from theoretical analysis.

4.3.2. Spoofing and Flooding

The network becomes sluggish, expensive, and complex due to the need for specialized intermediary equipment for traditional DDoS spoofing and scrubbing tactics. On the other hand, Afek et al.'s OpenFlow 1.5 and P4-based network spoofing and impersonation technologies provide a fresh method of thwarting SYN and DNS spoofing. This technique uses a high-speed programmable data plane to execute low-level operations at line speed. Moreover, efficiency is raised by scaling the quantity of controller messages and rules in accordance with the volume of valid traffic. Switches offer a more secure network environment by distributing flow tables, which helps safeguard big servers.

A method to lessen SYN flooding and ARP spoofing attacks is put forth by Lin et al. [95]. The suggested plan keeps track of SYN/ACK and ACK/FIN packet counts and alerts the controller to any anomalies. The attacker's IP address is appended to the packet reject rule by the controller. A software switch built on OpenFlow and P4 is implemented by Lin et al. [49] and its performance is compared in two scenarios. The P4-based solution may function in the data plane, which significantly decreases the corresponding data traffic, but the OpenFlow-based solution has a reasonably big throughput.

Paolucci et al. [96, 97] provide dynamic traffic engineering strategies for optical bypass and traffic reduction in response to TCP SYN flooding assaults. They also provide a P4-based stateful attack mitigation strategy. To document SYN flood assaults, data for every session are kept in the P4 log. Records are maintained for the number of SYN flood attempts and the port number of the last SYN packet. The packet is deleted if the quantity of detected attempts exceeds a certain threshold. The aforementioned functionalities were implemented by the authors on NetFPGA SUME cards and the bmv2 software key, respectively.

Musumeci et al. [98] combine maximum likelihood and P4 to mine data efficiently and provide a P4-based DDoS attack mitigation solution that uses ML classifier to fend against TCP flood attacks. In other words, the training and prediction times of several maximum likelihood estimation methods are compared by the authors in terms of accuracy and complexity. To extract the features utilized in the maximum likelihood classifier, P4 code is supplied. The characteristics are included in the metadata that is used to link extra data to the package. The learning outcomes dictate whether it is attacked or not.

4.4 Defense Integration

When it comes to performance efficiency, data plane computing may far outperform software-based applications. This boost in performance makes it possible to carry out high bandwidth and low latency operations in the data plane more efficiently. These operations may be carried out independently of the control plane thanks to certain algorithms that are exported to the data plane, which makes it possible to create more potent and responsive solutions. As a result, the stress on the control plane is lessened, network performance is enhanced overall, and crucial applications in data centers may run more effectively. With this technique, data centers can respond rapidly and function at peak efficiency even in situations with a lot of traffic.

4.4.1. Network-Level Defense

Xing et al. [99] propose Ripple, a programmable and distributed defense system to prevent link flooding at the network level for data centers. Link flooding in data centers is a common problem under high bandwidth demand and heavy traffic.

4.4.2. Deep Packet Inspection

In data centers, the increasing volume and hence complexity of network traffic requires more advanced techniques to ensure security and optimize network performance. In this context, Deep Packet Inspection (DPI) techniques are crucial to improve data center security and manage network traffic more efficiently.

NetWarden's hardware prototype can detect hidden network channels in data centers at full speed and reduce the impact of these channels with negligible performance impact. This system works smoothly in complex data center applications, increasing data security and minimizing performance loss. However, the latency of slow/fast path communication can be a disadvantage for some high-speed data center applications. Therefore, the effective implementation of systems such as NetWarden in data centers requires a comprehensive network security strategy and sufficient technical skills.

4.4.3. Machine Learning

The importance of using machine learning techniques to manage network traffic and ensure security in data centers is increasing day by day. Intelligent decision support systems in data centers analyze the characteristics of network traffic. This data is then processed using machine learning algorithms. By identifying previously undetected patterns and behaviors in network traffic in data centers, ML identifies security vulnerabilities and enables rapid response. In this way, network security can be significantly increased. Unlike traditional network security methods, ML-based security methods developed in the Software Defined Network (SDN) environment provide high accuracy and performance in data centers.

Moreover, it enables fast and effective processing of large amounts of data in data centers. However, implementing ML-based solutions in data centers requires technical skills and adequate infrastructure. In addition, a continuous improvement and upgrade process is required to optimize the performance of ML-based systems in data centers.

Laraba et al [110] used prolonged finite kingdom machine (EFSM) to version the kingdom-with the aid of using-kingdom protection tracking feature in statistics facilities. EFSMs can pick out greater complicated conditions and situations to reveal visitors and locate anomalies in statistics facilities. EFSMs outline states and variables the usage of a seven-detail index and may therefore specific greater complicated situations with fewer states. In statistics facilities, EFSMs are used to reveal visitors and locate anomalies. For example, a P4 key may be used to locate

Open Notification (ECN) conduct anomalies in statistics facilities. ECN is a mechanism used to file congestion situations in community visitors and is vital for visitors control in statistics facilities. By detecting such anomalies, EFSM can accurate the extraordinary conduct or reject the packet. This allows control community visitors in statistics facilities appropriately and efficiently.

In statistics facilities, FSMs play an critical position in handling community visitors and protection policies. FSMs reveal conversation among distinct gadgets at the community and course this conversation in step with positive rules. For example, FSMs may be used to make certain the enforcement of a specific protection coverage in statistics facilities. This way, community visitors is continuously monitored and coverage violation conduct is detected and prevented.

FSMs also can be used to control community visitors greater successfully in statistics facilities. For example, with the aid of using tracking visitors among distinct community gadgets in statistics facilities, FSMs can locate and block extraordinary visitors. This permits greater stable and green control of community visitors in statistics facilities.

There are many blessings to the usage of FSM in facts facilities. First, FSMs can speedy come across strange situations with the aid of using continuously tracking community site visitors. This will increase protection in facts facilities and stops ability vulnerability risks. Additionally, FSMs enhance community overall performance in facts facilities with the aid of using dealing with community site visitors greater effectively.

FSMs additionally make certain that protection regulations are enforced in facts facilities. This guarantees that community site visitors in facts facilities is routed in step with positive guidelines and that conduct that violates those guidelines is prevented. In this way, community site visitors in facts facilities is controlled in a greater steady and managed manner.

The use of FSM in facts facilities brings with it a few difficulties. First, FSMs need to be able to dealing with complicated community systems and site visitors situations. Additionally, FSMs want to be continuously up to date and optimized. This permits non-stop tracking of community site visitors in facts facilities and speedy detection of strange situations.

5. DIFFERENT P4 TARGETS FOR DATA CENTERS

Data centers are physical structures that store servers, cloud storage drives, network equipment, computing machines and related hardware equipment needed by the technologies we use in our digital world. These network structures face high performance, security, and flexibility requirements to provide the best service with the best efficiency.

P4, which allows programming protocol-independent packet processors, can be used to customize and optimize the behavior of network devices according to the need, and this makes P4 an important tool to meet the needs of data center networks in this area. Nearly half of the field's research has been conducted on software switches, which are supported by P4 in addition to hardware-based targets like NPU, FPGA, and ASIC [18].

In this section, we will compare the performance and concrete behavior of P4 for data centers in different targets for software and hardware.

• Using software to implement network protocols is flexible in terms of functionality and complexity, for example, BMv2 is a software switch simulator and can be used as a test and development tool by simulating programs written in P4 language on the CPU, but the processing capacity of the CPU limits the throughput and interrupt, and cache effects can affect delay and jitter [18][6].

• NPU is a type of software programmable ASIC optimized for network applications. They are part of a standalone network device or network card. For example, Netronome NFP-6000 smartNIC is an NPU-based smartNIC designed to accelerate network protocols and implement their functions. Compared to software switch, the throughput and latency performance of NPU is relatively improved, but flexibility is reduced [18][20].

• FPGA can be programmed to perform almost any function. For example, The FPGA-based PCI Express card NetFPGA SUME card, for instance, is suitable for use as a NIC, multiport switch, firewall, test/measurement environment, and more.[18] Apart from the limitation of memory resources, FPGAs generally exceed most targets in terms of throughput, jitter, and latency. However, FPGA programming requires specific hardware knowledge, and implementing network algorithms in HDL is very time-consuming [18][21].

• ASIC is superior in terms of throughput, i.e. throughput, latency, and jitter. For example, the Barefoot Tofino 2 is an ASIC-based network switch that offers flexibility, openness, and P4 programmability. However, the limited expressive power of the ASIC instruction set also limits the flexibility of function implementation [18][22].

The performance of different P4 targets is summarized in Table 7.

5.1 Performance Comparison

Performance is critical for high bandwidth and low latency requirements in data centers. The performance of different P4 targets is divided into throughput, latency, jitter and resource constraints in Table 7.

Table 7: Performance comparison of different p4 targets [18].

Metric	Software/CPU	NPU	FPGA	ASIC
Throughput	+	++	+++	++++
Latency	+ >10 us	++ 5 us ~ 10 us	+++ < 2 us	++++ < 2 us
Jitter	----	---	--	-
Resource constraints	++++	+++	++	+
Flexibility	++++	+++	++	+
Representative products	p4c-behavioral, bmv2, T4P4S	Netronome NFP-6000 SmartNIC [20], Pensando Capri [23]	NetFPGA SUME [21], P4FPGA [24]	Barefoot Tofino 1, Tofino 2 [5], Broadcom Tomahawk [25]
Proportion in the research (%)	48.5	5	7.9	38.6

5.2 Behavioral Comparison

The behavioral differences of P4 targets play an important role in determining the effects of the selected target on the security of data centers according to the target selection. While some P4 targets focus on basic packet processing and forwarding targets, others may provide more advanced features. For Data Centre security, capabilities such as packet filtering, encryption/decryption, deep packet inspection, and traffic engineering are important.

While some P4 targets provide higher-level abstractions and user-friendly interfaces, others may provide lower-level control and customization, and it is important for Data Centers to be able to customize the targets to meet their security requirements. The comparison of the concrete behaviors of different P4 targets is as shown in Table 8[18].

Table 8: Concrete behaviors towards different goals [18].

Purpose		CPU (bmv2)	FPGA (NetFPGA)	NPU (SmartNIC)	ASIC (Tofino)
[25] Implementing cryptographic hash function		Easy to implement, but with high latency (a few milliseconds)	Has the lowest latency but cannot be integrated using native P4 functions	Has the highest throughput, but cannot handle packets up to 900 bytes	——
[26] Impact of programs' bugs on network security	Reading invalid headers	Reading the flag value from the previous IPv4 packet	0 reading	——	Reading the value of the Explore Flags field
	Writing invalid titles	Writing and reading successfully	Writing and reading successfully	——	It can Write and read, but writing an invalid IPv4 header will cause the Explore header to change
	Loops	Resubmitting: blocks processing, discarding all subsequent packets	Unsupported	——	Resubmitting: no more than one resubmission; more resubmissions will be deleted; egress to egress clone: causing flooding
	Resurrecting dead packets	The field marked as "deprecated" can be used to store metadata which marks dropped packets	The packet whose egress metadata is set as the value of the egress port will be restored rather than discarded	——	Packets that do not specify a valid egress port are discarded by default
[27] Implementing SYN defense strategy		It cannot handle SYN flood traffic up to 14 Mpps	It can handle P4 programs with higher complexity; the delay is less than 10 μs, and there is no long tail; it only consumes one-third of the total resources	t processes data at almost linear speed; delay is between 1 and 4 μs;	——
[28] Implementing IP Sec		The delay between the P4 switch and the SDN controller is very low in experiment. Actually, the transmission delay is relatively large, covering the impact of key generation	(18) No support for parsing variable-length header fields. (23) Lack of data exchange between P4 pipeline and P4 extern	——	No support for user-defined P4 extern that contains computationally intensive functions. In this solution, P4 extern is relocated to CPU or directly forwards IPsec-related streams to the encryption host

Scholz et al. [25] address the need to implement cryptographic hash functions in the data plane to prevent attacks against potential hash collisions and propose a prototype implementation of cryptographic hash functions on three separate P4 targets (CPU, SmartNIC, and NetFPGA SUME). Table 2 shows the main evaluation results. The SmartNIC-based implementation has the highest throughput, but the CPU-based implementation offers the most flexibility overall. Users can implement the hash algorithm on different platforms depending on their performance requirements, depending on the platform they choose.

A first discussion of the extent to which the flaws can be exploited in practice is provided in [26]. First, to test three targets (Bmv2 switch, P4NetFPGA, and Tofino switch), the authors write a set of programs consisting of simple operations. The programming should consider the fact that the behavior of these targets may be faulty in certain cases to prevent extreme events. Then, if there are any exploitable vulnerabilities related to the NAT (Network Address Translation) function implemented in P4, they are examined. It is discovered that skilled attackers can perform attacks against various targets using simple NAT scheme and traditional switch scheme (switch.P4). Finally, the security implications of the above findings are investigated.

The use of their performance to defend the entire network against SYN flood attacks has been discussed by Scholz et al. [27]. Therefore, an analysis is conducted on two defense schemes: SYN authentication and SYN cookie. An in-depth discussion is given on the performance of these three targets and the implementation challenges encountered when porting them to software, FPGAs, and network processors.

5.3 P4 Target Selection for Data Center Security and Future Research

P4 target selection for data center security can be evaluated under the following headings: security requirements (what security functions are required and how strictly should they be implemented?), performance requirements (how much data center traffic and acceptable latency are there?), budget and resource constraints (hardware targets are generally more expensive but offer higher performance, whereas software targets may be more cost-effective and perform less well.) and current expertise (how experienced is the data center personnel profile against P4 targets and the current infrastructure?).

Future research can focus on the following topics on the role of P4 targets in Data Center security:

• New Security Functions: Discovering new security functions that can be implemented with P4 targets and evaluating the performance impacts of these security functions.

• Hybrid Targets: Developing new hybrid targets that can combine the advantages of Hardware and Software targets.

• Scalable Solutions and Standardization: Developing standardization solution proposals to increase the interoperability and portability of different P4 targets across different targets as the scale changes.

6. INNOVATIONS & CHALLENGES

There are several options to enhance data center network security with P4 programmable data planes. Past the prevailing patterns and obstacles, there exist many prospects for forthcoming investigations and execution of P4-oriented network security enhancements for data centers:

6.1 Innovations

6.1.1 Function Offloading

Significant advancements in network security are made possible by P4 programmable data planes. We can now move many tasks that were previously handled by the control plane to the data plane because of high-speed data processing capabilities and programmability. This process, referred to as "function offloading," reduces the strain on centralized SDN controllers and improves the scalability and efficiency of the network [30].

P4 programmable switches can take up numerous security duties that were previously managed by the control plane because of their capacity for packet processing. For example, P4 programmable switches may now carry out tasks directly in the data plane, such as packet filtering, access control lists (ACL) administration, and the execution of stateful security policies. This lessens the burden on the control plane and removes the central controller bottleneck effect [31].

Advantages of Function Offloading:

a. Quicker Reaction Times: Relocating security features to the data plane allows for quicker defense against network intrusions [33]. Attack mitigation and detection activities are carried out directly in the data plane, avoiding the need to go back and forth to the central controller, which reduces latency [32].

b. Enhanced Scalability: The network can accommodate more devices and higher traffic volumes due to a reduction in the strain on central controllers [34]. This guarantees the efficient maintenance of security features even in expansive networks.

c. Complex and Advanced Security Functions: P4 programmable data planes make it possible to implement more sophisticated security functions. The data plane can carry out certain encryption procedures as well as machine learning methods [35] for intrusion detection and deep packet inspection (DPI) [37].

d. Cost Savings: By having fewer central controllers, reducing function demand can save hardware costs. Energy can also be saved via a network architecture that is more effective [36].

6.1.2 Combining P4 with AI and Machine Learning

By incorporating machine learning algorithms into P4 programmable data planes [35], more intelligent and flexible approaches to identifying and thwarting assaults in data center networks may be provided. The creation of network security systems with the ability to self-heal, adapt, and identify unusual activity in real time may be made possible by this integration. P4 programmable switches [39], for instance, may use machine learning models to identify abnormalities in data center traffic [40] and react to them right away. This can make it possible to identify sophisticated and until undiscovered assaults in addition to well-known ones like zero-day attacks. Additionally, machine learning models can decrease false positives and improve the efficacy of network security systems by understanding typical traffic patterns in data centers.

6.1.3 Combination of P4 and Blockchain Technology

P4 programmable data planes and the safe, decentralized structure of blockchain technology may be coupled to create a new paradigm in data center network security. This can guarantee data integrity, make network infrastructures more transparent and safer, and make assaults easier to trace. Using a blockchain-based security mechanism [41], for instance, P4 programmable switches may verify the origin and integrity of every packet in the data center. Attackers may find it more difficult to deliver spoof packets or alter data center traffic as a result. Furthermore, blockchain technology can register security incidents in the data center, making it possible to identify and stop assaults faster [42].

6.1.4 Putting Zero Trust Networks into Practice with P4

Data centers are finding that the zero-trust paradigm is a crucial security measure. Because P4 programmable data planes allow zero trust regulations [37] to be applied directly at the network level, they may greatly increase the security of data centers. P4, for instance, allows for the dynamic application of distinct security policies in the data plane for every data stream. By obstructing unwanted access and stopping data breaches, this can improve the data center's security even further [43].

6.1.5 Data Center Segmentation using P4

Limiting security risks and halting the spread of assaults may be achieved by segmenting [41] data center networks into smaller, isolated portions. More flexibility and dynamic control over data center segmentation may be achieved using P4 programmable switches. For instance, P4 enables the creation of distinct segments for applications with various security needs, and P4 rules allow for the management of the traffic flow between these segments [44].

6.1.6 Hardware Improvements

More sophisticated and high-performing network security features may be possible with P4 target device hardware improvements. Faster and more memory capable P4 devices may operate better in data center network security applications. Faster P4 switches, for instance, can manage high-speed data center traffic more effectively, and larger memory capacities enable the implementation of more sophisticated security measures [45].

These trends may help to create next-generation data center security solutions and further uncover the potential of P4 programmable data planes in the field of data center network security.

6.2 Challenges:

6.2.1 Security Verification

Because the correctness of the data plane is not verified, programmability poses hidden risks even if it facilitates quick innovation and protocol creation. Inexperienced users frequently encounter this. By exploiting bugs in the software, attackers can launch an assault on the device. Developing automated testing and verification tools is critical to enhancing the programmable data plane's security. Because P4 applications use distinct programming objects [46], existing software testing tools cannot be used directly for P4 programs. Thus, a crucial area of future study will be the development and use of program analysis and verification tools for P4 [47].

6.2.2 Complex Calculations

Line-rate processing implies that each packet may manage fewer processing steps, and programmable switching only allows basic arithmetic operations for integer values. Loops can be used to enhance processing throughput; however, this comes at the cost of efficiency and is really just an overlay of simple processes rather than genuinely sophisticated calculations. The data plane's application scenarios are impacted by the absence of intricate computations [37].

6.2.3 Stateful Network Function:

The programmable data plane's [48] essential characteristic, state processing, makes it possible for a wide range of novel applications that are not feasible in non-programmable networks. This is the stateful network function. However, the size of the on-chip memory directly affects how much state data can be maintained since the programmable data plane needs to hold a certain amount of information. This limits the number of streams that can be processed and the operations that the data plane can carry out.

7. CONCLUSIONS

One potential way to strengthen data center security is via the P4 programmable data plane. The fundamentals of P4, recent findings, and its application to network security are reviewed in this article. P4 has proven that it can manage several security concerns, such as preventing unwanted access, safeguarding the privacy of data, and thwarting various assaults including distributed denial of service (DDoS).

Because of P4's adaptability and packet-level visibility, data centers may create unique solutions to overcome security flaws. Data centers may enhance their entire security posture and strengthen their defense against changing threats by utilizing P4. There are still issues, though, such the dearth of sophisticated calculations and the small number of stateful network services. Future research can concentrate on creating appropriate techniques for intricate computations and expanding stateful function storage to get over these obstacles. P4's ability to significantly improve data center security will become available with additional advancements in this field.

ACKNOWLEDGMENT

To the P4 community, we kindly thanks for their invaluable contributions to development and improvement of P4 technology. And, to all reference research, we kindly thanks them for improving the quality of this paper.

REFERENCES

[1] Website of the P4 Language Consortium, https://p4.org/.

[2] P4 Language Specification, https://p4.org/p4-spec/p4-14/v1.0.5/tex/p4.pdf.

[3] P4 Language Consortium, P4_16 Language Specification, P4.

[4] M. Shahbaz, S. Choi, P. Ben et al., "PISCES: A programmable, protocol-independent software switch," in Proceedings of the 2016 ACM SIGCOMM Conference, pp. 525–538, Florianopolis, Brazil, August 2016.

[5] N. Zilberman, Y. Audzevich, G. A. Covington, and A. W. Moore, "NetFPGA SUME: toward 100 Gbps as research commodity," IEEE Micro, vol. 34, no. 5, pp. 32–41, 2014.

[6] Behavioral model (bmv2), 2015, https://github.com/p4lang/ behavioral-model.

[7] Barefoot Tofino2,https://www.barefootnetworks.com/ products/brief-tofino-2/. plane-in-cha rge-of-theforwarding-plane.html.2017

[8] Debobroto Das Robin , Javed I. Khan , "P4TE: PISA switch based traffic engineering in fat-tree data center networks"

[9] N. McKeown, T. Sloane, and J. Wanderer, P4 Runtime-Putting the Control Plane in Charge of the Forwarding Plane, 2017, https://p4.org/api/p4-runtime-putting-the-control-

[10] Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, and David Walker. 2014. P4: Programming protocol-independent packet processors. SIGCOMM Comput. Commun. Rev. 44, 3 (July 2014), 87–95. https://doi.org/10.1145/2656877.2656890

[11] A. Yazdinejad, A. Bohlooli, and K. Jamshidi, "P4 to SDNet: Automatic generation of an efficient protocol-independent packet parser on reconfigurable hardware," in Proc. 8th Int. Conf. Comput. Knowl. Eng., 2018, pp. 159–164.

[12] D. Hancock and J. van der Merwe, "HyPer4: Using P4 to virtualize the programmable data plane," in Proc. 12th Int. Conf. Emerg. Netw. Exp. Technol. (CoNEXT), Irvine, CA, USA, 2016, pp. 35–49.

[13] C. Kozanitis, J. Huber, S. Singh, G. Varghese, Leaping multiple headers in a single bound: Wire-speed parsing using the Kangaroo system, in: 2010 Proceedings IEEE INFOCOM, IEEE, 2010, pp. 1–9.

[14] D. Franco, E. Ollora Zaballa, M. Zang, A. Atutxa, J. Sasiain, A. Pruski, E. Rojas, M. Higuero, and E. Jacob, "A comprehensive latency profiling study of the tofino P4 programmable asic-based hardware," Computer Communications, vol. 218, 2024.

[15] Soni, H., Rifai, M., Kumar, P., Doenges, R., and Foster, N. Composing dataplane programs with μp4. In ACM SIGCOMM (SIGCOMM) (2020).

[16] J. Gomez, E.F. Kfoury, J. Crichigno, G. Srivastava, A survey on TCP enhancements using P4-programmable devices, Comput. Netw. 212 (2022) 109030.

[17] Y.-J. Lin, C.-H. Hung, and C. H.-P. Wen, "Real-time in-network microburst mitigation on programmable switch," IEEE Access, vol. 10, pp. 2446–2456, 2022.

[18] Y. Gao, Z. Wang, A review of P4 programmable data planes for network security, Mob. Inf. Syst. 2021 (2021).

[19] Netronome Agilio CX SmartNICs, 2019, https://netronome.com/agilio-smartnics/.

[20] NetFPGA SUME, https://netfpga.org/NetFPGA-SUME.html.

[21] BarefootTofino2, https://www.barefootnetworks.com/products/brief-tofino-2/.

[22] Pensando Distributed Services Architecture SmartNIC, https://www.servethehome.com/pensando-distributed-services-architecture-smartnic/.

[23] H. Wang, R. Soul´e, H. Tu Dang et al., "P4FPGA: a rapid prototyping framework for P4," in Proceedings of the ACM Symposium on SDN Research (SOSR), Santa Clara, CA, USA, 2017.

[24] Tomahawk/BCM56960Series, https://www.broadcom.com/products/ethernet-connectivity/switching/strataxgs/bcm56960-series.

[25] D. Scholz, A. Oeldemann, F. Geyer et al., "Cryptographic hashing in P4 data planes," in Proceedings of the 2019 ACM/ IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), pp. 1–6, IEEE, Cambridge, UK, September 2019.

[26] M. Valentin Dumitru, D. Dumitrescu, and C. Raiciu, "Can we exploit buggy P4 programs?" in Proceedings of the Symposium on SDN Research, San Jose, CA, USA, 2020.

[27] D. Scholz, S. Gallenm¨uller, H. Stubbe, B. Jaber, M. Rouhi, and G. Carle, "Me love (SYN-) cookies: SYN flood mitigation in programmable data planes," 2020, https://arxiv.org/abs/2003.03221.

[28] F. Hauser, M. H¨aberle, M. Schmidt, and M. Menth, "P4-IPsec: SitetoSite and host-to-site VPN with IPsec in P4-based SDN," IEEE Access, vol. 8, 2020.

[29] Zilberman, Noa & Audzevich, Yury & Covington, G. & Moore, Andrew. (2014). NetFPGA SUME: Toward 100 Gbps as research commodity. Micro, IEEE. 34. 32-41. 10.1109/MM.2014.61.

[30] D. Moro, M. Peuster, H. Karl, and A. Capone, "FOP4: Function Offloading Prototyping in Heterogeneous and Programmable Network Scenarios," in 2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Dallas, TX, USA, 2019, pp. 1-6, doi: 10.1109/NFV-SDN47374.2019.9040052.

[31] P4sec authors. "P4sec: A Security-Enhanced SDN Architecture with P4 Programmable Switches." [Online]. Available: GitHub.

[32] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming protocol-independent packet processors," in ACM SIGCOMM Computer Communication Review, vol. 44, no. 3, pp. 87-95, 2014.

[33] Y. Bremler-Barr, E. Levy-Abegnoli, and Y. Elovici, "In-Network Intrusion Detection and Prevention Systems for SDN," Security and Communication Networks, vol. 9, no. 18, pp. 5565-5580, Dec. 2016. doi:10.1002/sec.1773.

[34] G. Gibb, G. Varghese, and N. McKeown, "Flexible and scalable network security with P4," 2016 IEEE 24th International Conference on Network Protocols (ICNP), Singapore, 2016, pp. 1-10.

[35] X. Jin, X. Li, H. Zhang, R. Yu, J. Liu, and Y. Xiang, "P4ML: A P4-Based Deep Learning Framework for Programmable Data Planes," in IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, Toronto, ON, Canada, 2020, pp. 1699-1708.

[36] K. Bu, X. Wen, Y. Chen, L. Huang, and X. Gao, "Energy-Efficient Data Plane Programming for Network Functions," IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 3, pp. 693-705, March 2021.

[37] A. Sivaraman, A. Cheung, M. Tan, and G. Varghese, "Packet Transactions: High-Level Programming for Line-Rate Switches," Proceedings of the 2016 ACM SIGCOMM Conference, Florianopolis, Brazil, 2016, pp. 15-28. doi: 10.1145/2934872.2934900.

[38] P4lang. "p4app-switchML: Switch ML Application." GitHub. Available: https://github.com/p4lang/p4app-switchML

[39] A. Sgambelluri, A. Giorgetti, F. Cugini, L. Valcarenghi, and P. Castoldi, "Demonstration of P4 Neural Network Switch," in Optical Fiber Communication Conference, 2021, Paper W1I.5. doi:10.1364/OFC.2021.W1I.5.

[40] M. C. Luizelli, L. Z. Granville, G. B. Oliveira, M. S. Santos, R. C. Almeida, and R. de Sousa, "Machine-Learning-Enabled DDoS Attacks Detection in P4 Programmable Networks," in 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 2018, pp. 1042-1047.

[41] Y. Zhang, Y. Zhang, S. Chen, and W. Zhou, "P4Block: A Blockchain-Based Architecture for P4 Programmable Data Planes," 2022 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2022, pp. 1522-1529.

[42] S. Fu, Y. Wen, X. Huang, F. R. Yu, and H. Zhang, "Leveraging Blockchain and SDN for Security in 5G and Beyond Networks: A Comprehensive Survey," IEEE Communications Surveys & Tutorials, vol. 24, no. 2, pp. 775-805, Secondquarter 2022.

[43] E. Ollora and D. Franco, "P4Knocking: Offloading Host-Based Firewall Functionalities to the Network," in 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2020, pp. 303-309.

[44] D. Han, J. Lee, H. Kim, K. Lim, and T. Lee, "P4Visor: A P4-Based Architecture for Data Center Network Visibility and Analytics," in 2018 IEEE 26th International Conference on Network Protocols (ICNP), Cambridge, United Kingdom, 2018, pp. 259-270.

[45] S. Ibanez, G. Brebner, N. McKeown, and N. Zilberman, "P4 programmable switches: Recent advances and future challenges,"

IEEE Journal on Selected Areas in Communications, vol. 38, no. 7, pp. 1475-1491, July 2020.

[46] F. Freire, M. Barcellos, L. Rech, R. Reis, and P. Fonseca, "Finding Vulnerabilities in P4 Programs with Assertion-based Verification," in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security Posters, Dallas, Texas, USA, 2017, pp. 563-565.

[47] C. J. Anderson, N. Foster, A. Guha, J. P. Kohler, C. Schlesinger, and D. Walker, "Automatically verifying reachability and well-formedness in P4 Networks," in Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI '15), Oakland, CA, USA, 2015, pp. 357-370.

[48] L. Jose, L. Yan, G. Varghese, and N. McKeown, "Compiling Packet Programs to Reconfigurable Switches," in Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI '15), Oakland, CA, USA, 2015, pp. 371-384.

[49] M. U. Hassan, M. H. Rehmani, and J. H. Park, "Security and privacy challenges in software-defined networks: a comprehensive survey," Mobile Information Systems, vol. 2021, Article ID 1257046, 2021.

[50] P. Vörös and A. Kiss, "Security middleware programming using P4," in Proceedings of the International Conference on Human Aspects of Information Security, Privacy, and Trust (HAS), Vancouver, BC, Canada, July 2016.

[51] J. Cao, J. Bi, Y. Zhou, and C. Zhang, "CoFilter: a high-performance switch-assisted stateful packet filter," in Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos, Budapest, Hungary, August 2018.

[52] J. Deng, H. Hu, H. Li et al., "VNGuard: an NFV/SDN combination framework for provisioning and managing virtual firewalls," in Proceedings of the IEEE Conference on Network Function Virtualization and Software-Defined Networking (NFV-SDN), San Francisco, CA, USA, 2015.

[53] R. Datta, S. Choi, A. Chowdhary, and Y. Park, "P4Guard: designing P4 based firewall," in Proceedings of the IEEE Military Communications Conference (MILCOM), Los Angeles, CA, USA, October 2018.

[54] R. Ricart-Sanchez, M. Pedro, J. M. Alcaraz-Calero, and Q. Wang, "NetFPGA-based firewall solution for 5G multi-tenant architectures," in Proceedings of the IEEE International Conference on Edge Computing (EDGE), Milan, Italy, July 2019.

[55] A. Almaini, A. Al-Dubai, I. Romdhani, and M. Schramm, "Delegation of authentication to the data plane in software-defined networks," in Proceedings of the 2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS), Shenyang, China, 2019.

[56] E. O. Zaballa, D. Franco, Z. Zhou, and M. S. Berger, "P4Knocking: offloading hostbased firewall functionalities to the network," in Proceedings of the 2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), Paris, France, February 2020.

[57] A. Almaini, A. Al-Dubai, I. Romdhani, M. Schramm, and A. Alsarhan, "Lightweight edge authentication for software defined networks," Computing, vol. 103, no. 2, pp. 291–311, 2021.

[58] K. Qiao, L. Xue, M. Adam, Y. Tang, A. Chen, and X. Luo, "Programmable in-network security for context-aware BYOD policies," in Proceedings of the USENIX Security Symposium, Boston, MA, USA, August 2020.

[59] S. Bai, H. Kim, and J. Rexford, "Passive OS fingerprinting on commodity switches," in Proceedings of the CoNEXT'19, Orlando, FL, USA, December 2019.

[60] Y.-Z. Cai, T.-Y. Lin, Y.-T. Wang, Y.-P. Tuan, and M.-H. Tsai, "EReplacement: efficient scanner data collection method in P4-based software-defined networks," International Journal of Network Management, vol. 31, no. 6, p. e2162, 2021.

[61] D. Scholz, S. Gallenmüller, Henning Stubbe, and G. Carle, "SYN flood defense in programmable data planes," in Proceedings of the 3rd P4 Workshop in Europe, Barcelona, Spain, December 2020.

[62] H. Gondaliya, G. C. Sankaran, and K. M. Sivalingam, "Comparative evaluation of IP address anti-spoofing mechanisms using a P4/NetFPGA-based switch," in Proceedings of the P4 Workshop in Europe (EuroP4), Barcelona, Spain, December 2020.

[63] K. Peng, Y. Liu, and H. Lin, "P4DAD: securing duplicate address detection using P4," in Proceedings of the IEEE International Conference on Communications (ICC), Montreal, QC, Canada, June 2020.

[64] G. K. Ndonda and R. Sadre, "A two-level intrusion detection system for industrial control system networks using P4," in Proceedings of the International Symposium for ICS & SCADA Cyber Security Research (ICS-CSR), Hamburg, Germany, August 2018.

[65] B. Lewis, M. Broadbent, and N. Race, "P4ID: P4 enhanced intrusion detection," in Proceedings of the IEEE Conference on Network Function Virtualization and Software-Defined Networking (NFV-SDN), Dallas, TX, USA, November 2019.

[66] A. Sanghi, K. P. Kadiyala, P. Tammana, and S. Joshi, "Anomaly detection in data plane systems using packet execution paths, SPIN'21," in Proceedings of the ACM SIGCOMM 2021 Workshop on Secure Programmable network INfrastructure, pp. 9–15, Vienna, Austria, December 2021.

[67] R. Meier, P. Tsankov, V. Lenders, L. Vanbever, and M. Vechev, "NetHide: secure and practical network topology obfuscation," in Proceedings of the 27th{USENIX} Security Symposium, pp. 693–709, Baltimore, MD, USA, August 2018.

[68] T. Datta, N. Feamster, J. Rexford, and L. Wang, "SPINE: surveillance protection in the network Elements," in Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI), Santa Clara, CA, USA, August 2019.

[69] D. Chang, W. Sun, and Y. Yang, "A SDN proactive defense mechanism based on IP transformation," in Proceedings of the International Conference on Safety Produce Informatization (IICSPI), Chongqing, China, November 2019.

[70] Y. Qin, W. Quan, F. Song et al., "Flexible encryption for reliable transmission based on the P4 programmable platform," in Proceedings of the 2020 Information Communication Technologies Conference (ICTC), Nanjing, China, May 2020.

[71] Y. Govil, L. Wang, and J. Rexford, "MIMIQ - masking IPs with migration in QUIC," in Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI), Austin, TX, USA, August 2020.

[72] M. Liu, D. Gao, G. Liu et al., "Learning based adaptive network immune mechanism to defense eavesdropping attacks," IEEE Access, vol. 7, pp. 182814–182826, 2019.

[73] G. Liu, W. Quan, N. Cheng, N. Lu, H. Zhang, and X. Shen, "P4NIS: improving network immunity against eavesdropping with programmable data planes," in Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, July 2020.

[74] C. Zhou, W. Quan, D. Gao et al., "AMS: adaptive multipath scheduling mechanism against eavesdropping attacks with programmable data planes," in Proceedings of the IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 2021.

[75] H. Mohajeri Moghaddam and A. Mosenia, "Anonymizing masses: practical light-weight anonymity at the network level," 2019, https://arxiv.org/abs/1911.09642.

[76] X. Chen, "Implementing AES encryption on programmable switches via scrambled lookup tables," in Proceedings of the Proceedings of the Workshop on Secure Programmable Network Infrastructure, 2020.

[77] F. Hauser, M. Häberle, M. Schmidt, and M. Menth, "P4-IPsec: SitetoSite and host-to-site VPN with IPsec in P4-based SDN," IEEE Access, vol. 8, 2020.

[78] F. Hauser, M. Schmidt, M. Häberle, and M. Menth, "P4-MACsec: dynamic topology monitoring and data layer protection with MACsec in P4-based SDN," IEEE Access, vol. 8, 2020.

[79] H. Kim and A. Gupta, "ONTAS: flexible and scalable online network traffic anonymization system," in Proceedings of the 2019 Workshop on Network Meets AI & ML - NetAI'19, Beijing China, August 2019.

[80] Z. Zhibin, C. Chang, and X. Zhu, "A software-defined networking packet forwarding verification mechanism based on programmable data plane," Journal of Electronics and Information Technology, vol. 42, no. 5, pp. 1110–1117, 2020.

[81] G. Grigoryan and Y. Liu, "LAMP: prompt layer 7 attack mitigation with programmable data planes," in Proceedings of the 2018 IEEE

17th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, November 2018.

[82] A. Febro, H. Xiao, and J. Spring, "Telephony denial of service defense at data plane (TDoSD@DP)," in Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS), Taipei, Taiwan, April 2018.

[83] A. Febro, H. Xiao, and J. Spring, "Distributed SIP DDoS defense with P4," in Proceedings of the 2019 IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–8, IEEE, Marrakech, Morocco, April 2019.

[84] M. Kuka, K. Vojanec, K. Jan, and P. Benáček, "Accelerated DDoS attacks mitigation using programmable data plane," in Proceedings of the ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), Cambridge, UK, September 2019.

[85] J. Ioannidis and S. M. Bellovin, "Implementing pushback: router-based defense against DDoS attacks," in Proceedings of the NDSS, San Diego, CA, USA, February 2016.

[86] M. Yu and A. Wang, "ML-Pushback," in Proceedings of the 15th International Conference on emerging Networking Experiments and Technologies, Orlando, FL, USA, December 2019.

[87] L. A. Quintero Gonz'alez, L. Castanheira, J. A. Marques, A. Schaeffer-Filho, and L. P. Gaspary, "BUNGEE: an adaptive pushback mechanism for DDoS detection and mitigation in P4 data planes," in Proceedings of the 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 393–401, Bordeaux, France, May 2021.

[88] Â. Cardoso Lapolli, J. A. Marques, and L. P. Gaspary, "Offloading real-time DDoS attack detection to programmable data planes," in Proceedings of the IFIP/IEEE Symposium on Integrated Management (IM), Washington, DC, USA, April 2019.

[89] X. Z. Khooi, L. Csikor, D. M. Divakaran, and M. S. Kang, "DIDA: distributed in-network defense architecture against amplified reflection DDoS attacks," in Proceedings of the 2020 6th IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, July 2020.

[90] M. Dimolianis, P. Adam, and V. Maglaris, "A multi-feature DDoS detection schema on P4 network hardware," in Proceedings of the 2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), Paris, France, February 2020.

[91] F. Kurt, E. Kfoury, E. Bou-Harb, and J. Crichigno, "Towards a unified in-network DDoS detection and mitigation strategy," in Proceedings of the 2020 6th IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, July 2020.

[92] Goksel Simsek, H. Bostan, A. K. Sarica et al., "Drop: a P4 approach to mitigating dos attacks in SDN," in Proceedings of the International Workshop on Information Security Applications, pp. 55–66, Springer, Thanjavur, India, November 2019.

[93] D. Ding, M. Savi, F. Pederzolli, M. Campanella, and D. Siracusa, "In-network volumetric DDoS victim identification using programmable commodity switches," 2021, https://arxiv.org/abs/2104.06277v3.

[94] Y. Afek, A. Bremler-Barr, and L. Shafir, "Network anti-spoofing with SDN data plane," in Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), Atlanta, GA, USA, May 2017.

[95] T.-Y. Lin, J.-P. Wu, P.-H. Hung et al., "Mitigating SYN flooding attack and ARP spoofing in SDN data plane," in Proceedings of the 2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS), Daegu, Korea, September 2020.

[96] F. Paolucci, F. Cugini, and P. Castoldi, "P4-based multi-layer traffic engineering encompassing cyber security," in Proceedings of the Optical Fiber Communication Conference (OFC), San Diego, CA, USA, March 2018.

[97] S. Yang, B. Lu, L. Cui et al., "An efficient pipeline processing scheme for programming Protocol-independent Packet Processors," Journal of Network and Computer Applications, vol. 171, 2020.

[98] F. Musumeci, V. Ionata, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-assisted DDoS attack detection with P4 language," in Proceedings of the IEEE International Conference on Communications (ICC), Montreal, QC, Canada, July 2020.

[99] Xing J., Wu W., and Chen A., Ripple: a programmable, decentralized link-flooding defense against adaptive adversaries, Proceedings of the 30th USENIX Security Symposium, 2021, Vancouver, Canada.

[100] Zhang M., Li G., Wang S., Chang L., Chen A., Hu H., Gu G., Qi L., Xu M., and Wu J., Poseidon: mitigating volumetric DDoS attacks with programmable switches, Proceedings of the NDSS, June 2020, San Diego, CA, USA.

[101] Liu Z., Hun N., Nikolaidis G., Lee J., Kim C., Jin X., Braverman V., Yu M., and Sekar V., Jaqen: a high-performance switch-native approach for detecting and mitigating volumetric DDoS attacks with programmable switches, Proceedings of the 30th {USENIX} Security Symposium, August 2021, Vancouver, BC, Canada, 3829–3846.

[102] Xing J., Wu W., and Chen A., Architecting programmable data plane defenses into the network with FastFlex, Proceedings of the 18th ACM Workshop on Hot Topics in Networks, November 2019, Princeton, NJ, USA.

[103] Hypolite J., Sonchack J., Hershkop S., Nathan D., DeHon A., and Jonathan M. S., DeepMatch: practical deep packet inspection in the data plane using network processors, Proceedings of the ACM Conference on emerging Networking Experiments and Technologies (CoNEXT), 2020, Barcelona, Spain.

[104] Li G., Zhang M., Chang L., Kong X., Chen A., Gu G., and Duan H., NetHCF: enabling line-rate and adaptive spoofed IP traffic filtering, Proceedings of the 2019 IEEE 27th International Conference on Network Protocols (ICNP), October 2019, Chicago, IL, USA, IEEE, 1–12.

[105] Xing J., Qiao K., and Chen A., NetWarden: mitigating network covert channels while preserving performance, Proceedings of the 29th {USENIX} Security Symposium, August 2020, San Diego, CA, USA.

[106] Xie J., Richard Yu F., Huang T., Xie R., Jiang L., and Wang C., A survey of machine learning techniques applied to software defined networking (SDN): research issues and challenges, IEEE Communications Surveys & Tutorials. (2018) 21, no. 1, 393–430.

[107] Qin Q., Poularakis K., Kin K. L., and Tassiulas L., Line-speed and scalable intrusion detection at the network edge via federated learning, Proceedings of the IFIP-TC6 Networking Conference (Networking), May 2020, Valencia, Spain.

[108] Barradas D., Santos N., Rodrigues L., Signorello S., Fernando M., Ramos V., and Madeira A., FlowLens enabling efficient flow classification for ML-based network security applications, Proceedings of the Network and Distributed Systems Security (NDSS) Symposium, August 2021, San Diego, CA, USA.

[109] Gutiérrez S. A., Branch J. W., Gaspary L. P., and Botero J. F., Watching smartly from the bottom intrusion detection revamped through programmable networks and artificial intelligence, 2021.

[110] Laraba A., François J., Chrisment I., Chowdhury S. R., and Boutaba R., Defeating protocol abuse with P4: application to explicit congestion notification, Proceedings of the IFIP-TC6 Networking Conference (Networking), May 2020, Valencia, Spain.